MERGER board (CREAM experiment)

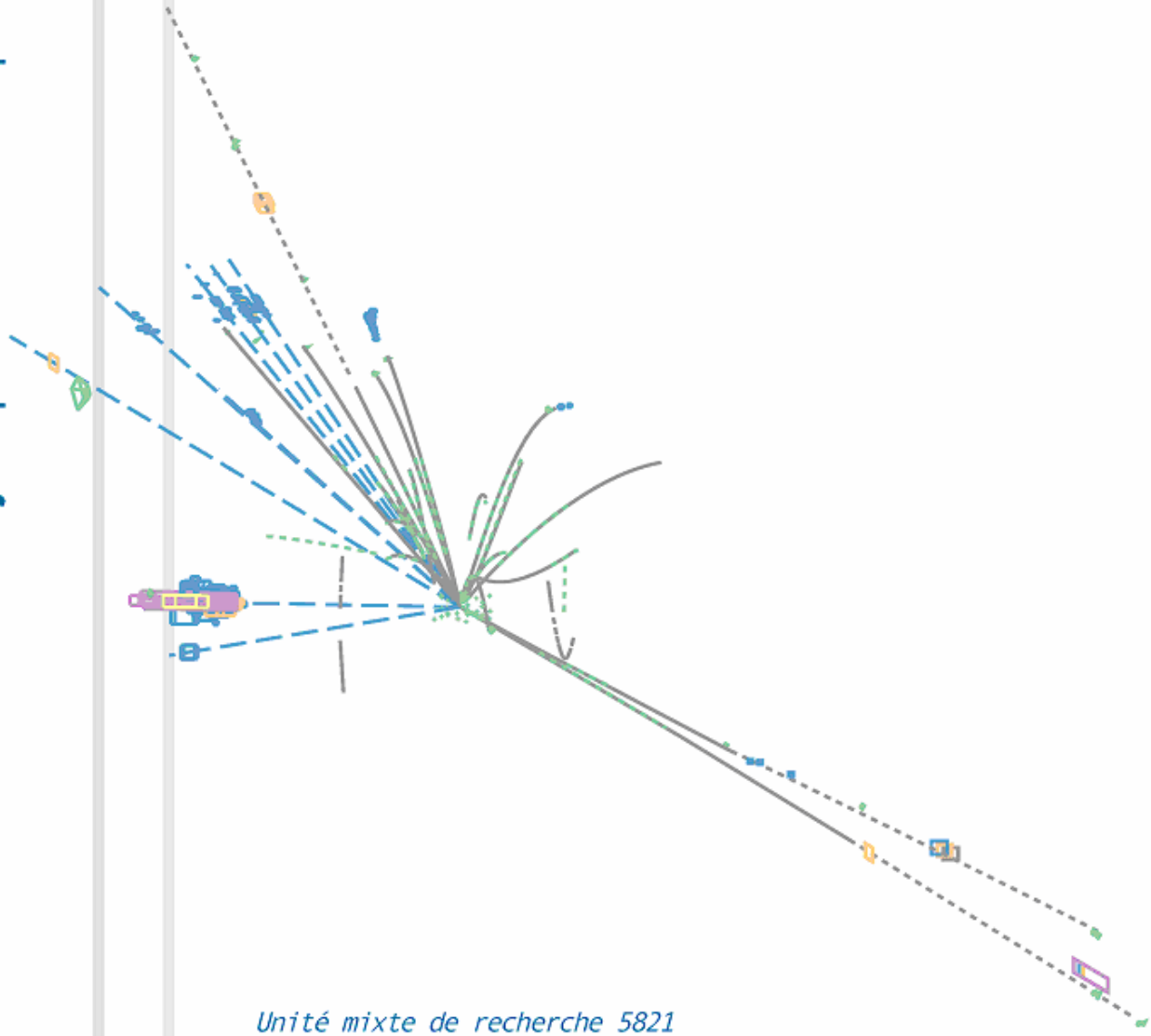
J. Bouvier

Internal report LPSC 07-48
Version 8
May 2007

MERGER board
(CREAM experiment)

Serializer board of the data PMT matrix for the CHERCAM detector

Joël BOUVIER
Data Acquisition Team



Laboratoire de Physique Subatomique et de Cosmologie

Unité mixte de recherche 5821
CNRS-IN2P3 / UJF / INPG

1. Overview

The purpose of this board is to interface the DAQ_FEE boards, board located behind the PMT, which assumes the conversion of the analog value issue by the PMT, and the SPARSIFICATION board, which assumes the interface with the CREAM balloon.

Its main tasks are:

- ✓ Manage the power voltage for the DAQ_FEE boards
- ✓ Transfer the command from the SRPARSIFICATION board to the DAQ_FEE boards
- ✓ Manage the data transfer from the DAQ_FEE boards to the MERGER board (serial acquisition data)
- ✓ Serial to parallel conversion for the data to the SPARSIFICATION board

2. Block Diagram

2.1. Logical parts block diagram

A block diagram concerning the logical parts of the design are shown in Annex 1 : Logical parts block diagram with 21 bits SERDES page 22

2.2. Power parts block diagram

A block diagram concerning the power parts of the design are shown in Annex 2 : Power parts block diagram page 23

3. Data acquisition command description

The following description is available for the command bus from the SPARSIFICATION board and to the DAQ_FEE boards.
For most command the MERGER board behaves like a command repeater to the DAQ_FEE board.

The command is sent via a serial link and validated by an enable signal (low level active).
The number of signals necessary for this function is equal to 3 :

- ✓ A clock signal (CK40)
- ✓ Data signal (DATA)
- ✓ A Strobe signal (/ENABLE)

Two command types are defined:

- ✓ Command with argument
- ✓ Command without argument

For safety reason no commands are defined with a code equal to 0xFF (hexadecimal representation) because if, for any reasons, the data line becomes unconnected the data view by the receptor is a continuous high level logic.

The Data and Strobe signals are generated on the rising edge of the Clock signal and the most significant bit of the coded commands is always sent first.

3.1. Command with argument

This command type is used for initializing a value into the DAQ_FEE board or into the MERGER board. All the command of this type is issue by the SPARSIFICATION board.

The command is coded on 16 bits word and a timing diagram relative to this function is showing on the Figure 1 : command with argument.

the most significant bit is sent first.

Its structure is :

- ✓ The first 8 bits is for the command definition
- ✓ The others for the data value.

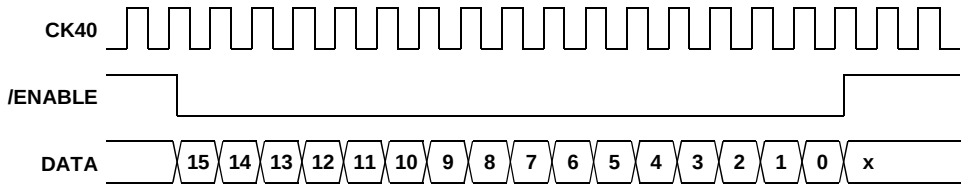
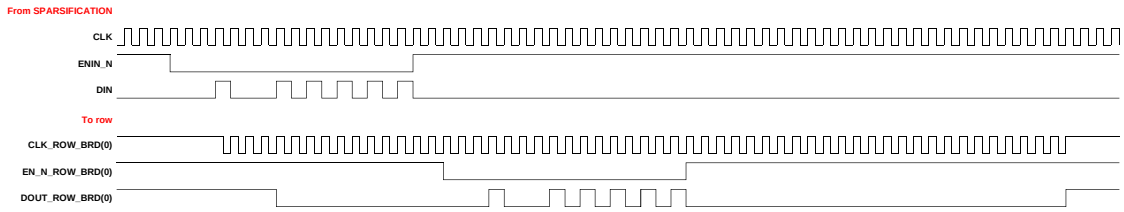


Figure 1 : command with argument



For more information about the command with argument refer to the SPARSIFICATION box documentation.

The declared command is :

- ✓ **Initialization of the DAQ_FEE internal delay :**

  This command type is for initialize the delay between the TRIGGER command and the start of the analog conversion.
  There is one delay value per DAQ_FEE board.
  The command is defined like below :
  - Bit(15..12) represents the column number. The numbers available are **0 to 4** and **8 to C** (hexadecimal value).
  - Bit(11..8) represents the row number. The numbers available are **0 to 4** and **8 to C** (hexadecimal value).
  - Bit(7..0) represents the data value (for more information about the description of this value refer to the DAQ_FEE documentation)

- ✓ **Initialization of the MERGER internal register :**

  This command type is for initialize the MERGER internal register. This one are also sent to the DAQ_FEE which not interpreted them.
  By default, at the power up or after a RESET state all the MERGER internal register have a zero value.
  The command is defined like below :
  - Bit(15..12) **D** (hexadecimal value). This value represents a command to the MERGER internal register.
  - Bit(11..8) represents the address register which is to be initialized. (16 possible value)
  - Bit(7..0) represents the data which be writing into the internal register.

The declared registers are :

<table>
<tr><th>Address</th><th>Name</th><th>Designation</th></tr>
<tr><td>0</td><td>LED_VALUE_LSB</td><td rowspan="2">This register represents the LED intensity value used for the test of the matrix.
The low significant bit is equal to xx mV.
The assignment register is like the following description :
D[15..14] : don’t care
D[13..12] : 0 0 Normal operation
0 1 1 K□ to Gnd
1 0 100 K□ to gnd
1 1 Three state
the last 3 values are Power down modes
D[11..0] : Data to convert
By default this 16 bit register is set to 0.
For more information’s see the AD5328 datasheet component.</td></tr>
<tr><td>1</td><td>LED_VALUE_MSB</td></tr>
<tr><td>2</td><td>LED_ON</td><td>This command indicates to the MERGER board that is entering in a test mode.
For the following Acquisition Trigger command (x50) receiving from the SPARSIFICATION board it must generate this sequence of action :
✓ generated a LED pulse command
✓ transmit the Acquisition Trigger command to the DAQ_FEE boards</td></tr>
<tr><td>3</td><td>LED_OFF</td><td>This command indicates to the MERGER board that is entering in a standard mode.
For the following Acquisition Trigger command (x50) receiving from the SPARSIFICATION board it must generate this sequence of action :
✓ transmit the Acquisition Trigger command to the DAQ_FEE boards</td></tr>
</table>

The Annex 4 shown show the functionality of functions LED_ON and LED_OFF.

If an function need more than one register (ex. 16 bit register) the global value is transferred to the function when the More Significant Byte (MSB) is received.

3.2. Command without argument

This command type is used for driving the DAQ_FEE board or the MERGER board. Most of the commands are issue by the SPARSIFICATION board however the MERGER board can also send some commands to the DAQ_FEE board.

The command is coded on 8 bits word and a timing diagram relative to this function is showing on the Figure 2.

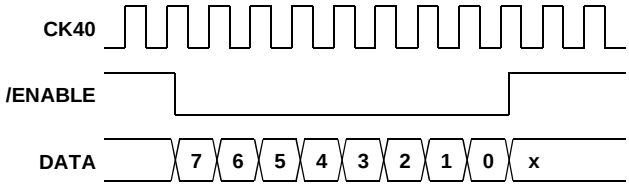
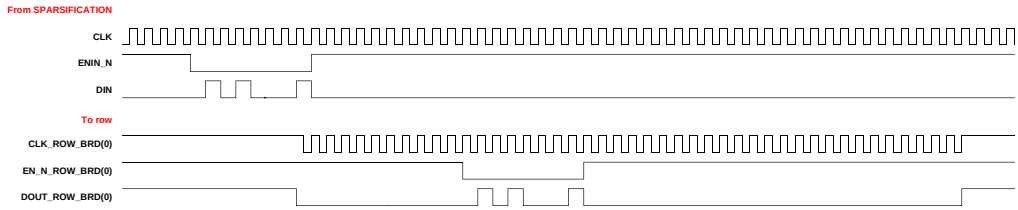


Figure 2 : command without argument



The declared command is:

- ✓ **Acquisition Trigger**[1] : 0x50
  This command is used when a trigger is generated by the master trigger; this command comes from the SPARSIFICATION board.

- ✓ **Calibration gain 1**[1] : 0x51
  This command is used to indicate to the DAQ_FEE boards that the next Acquisition trigger must be treated like CALIBRATION GAIN 1. This command comes from the SPARSIFICATION board.

- ✓ **Calibration gain 5**[1] : 0x52
  This command is used to indicate to the DAQ_FEE boards that the next Acquisition trigger must be treated like CALIBRATION GAIN 5. This command comes from the SPARSIFICATION board.

- ✓ **Normal aquisition**[1] : 0x53
  This command is used to indicate to the DAQ_FEE boards that the next Acquisition trigger must be treated like NORMAL ACQUISITION. This command comes from the SPARSIFICATION board.

- ✓ **Enable To Transmit Data** : 0x54
  This command is used to ask at the DAQ_FEE board to send their acquired data's. This command is sent by the MERGER board to the row one after the others with a delay after it have received a Acquisition Trigger command.

[1] For more information about this command see the SPARSIFICATION board documentation

4. Data acquisition reading

4.1. Read data principle

4.1.1. General read data principle

The reading principle is to acquire data's, for one MERGER board, one half of the matrix data row by row the second half being made by the 2nd MERGER board.
The following figure shows the relative position of the MERGER boards and the detector.

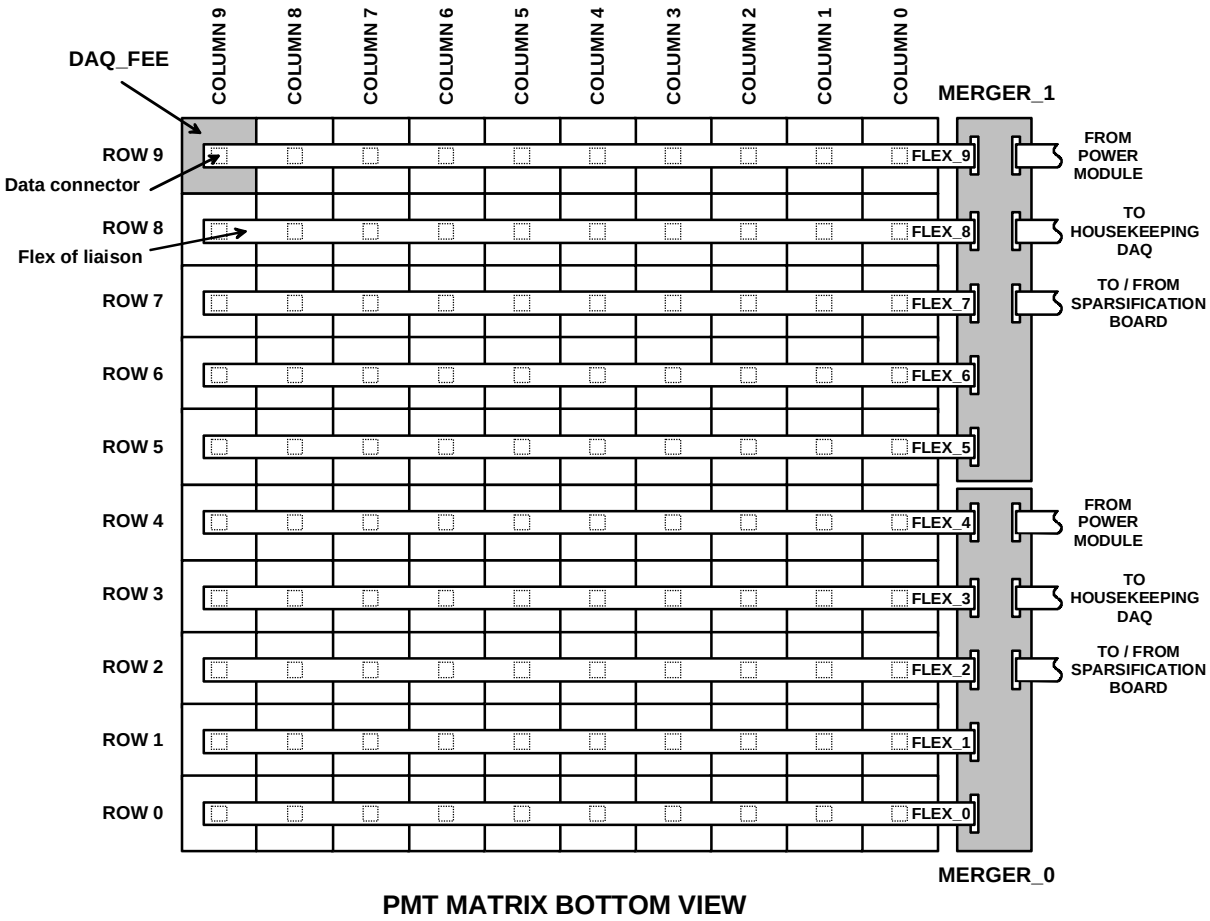


Figure 3 : DAQ matrix implementation

The acquisition of the complete matrix needs 2 MERGER boards which work in parallel and independently.
The following functional description is for one MERGER board, the second being the same functionally.
During the acquisition of a row, the 4 others are in idle mode. This mode is obtained by putting the clock to the DAQ_FEE board in a fixed state. This operation is doing by disabling the clock driver component; previously the DAQ_FEE board has entered in a stable state when it finished the last receiving command (for more information see the documentation of the DAQ_FFE board and the Data acquisition command description chapter in the present documentation).
So the clock lines on the disabled flexes are in a three state which is seen by the receiver component on the DAQ_FEE board like a high level logic[2] (for more information see the

[2] A fail-safe feature sets the output high when the inputs are open, or when the inputs are undriven and shorted or undriven and parallel terminated (extract of the MAX9173 documentation, Quad LVDS Line Receiver with Flow-Through Pinout and "In-Path" Fail-Safe from MAXIM)

documentation about the LVDS receiver).

4.1.2. Row read data principle

For a row acquisition, each DAQ_FEE board has 16 values to be transmitted to the SPARSIFICATION board. After receiving an **Enable To Transmit Data command**, each DAQ_FEE board transmit their data on their dedicated data serial line like the timing determined in the Annex 6 : Row Data Acquisition Sequence without gap between 2 words. The most significant bit of a value is always sent first and the low significant value of a DAQ_FEE board is also always sent first.
When all the word (10 x 18 bits) of a specific index (for example all the Value_0 of the row) are stored in the MERGER board, they are retransmitted, in parallel and in sequence, to the SPARSIFICATION board while the data of the immediately superior index is send from the DAQ_FEE boards.
The configuration of the different index is shown in the following figure (Figure 4 : Group words definition).

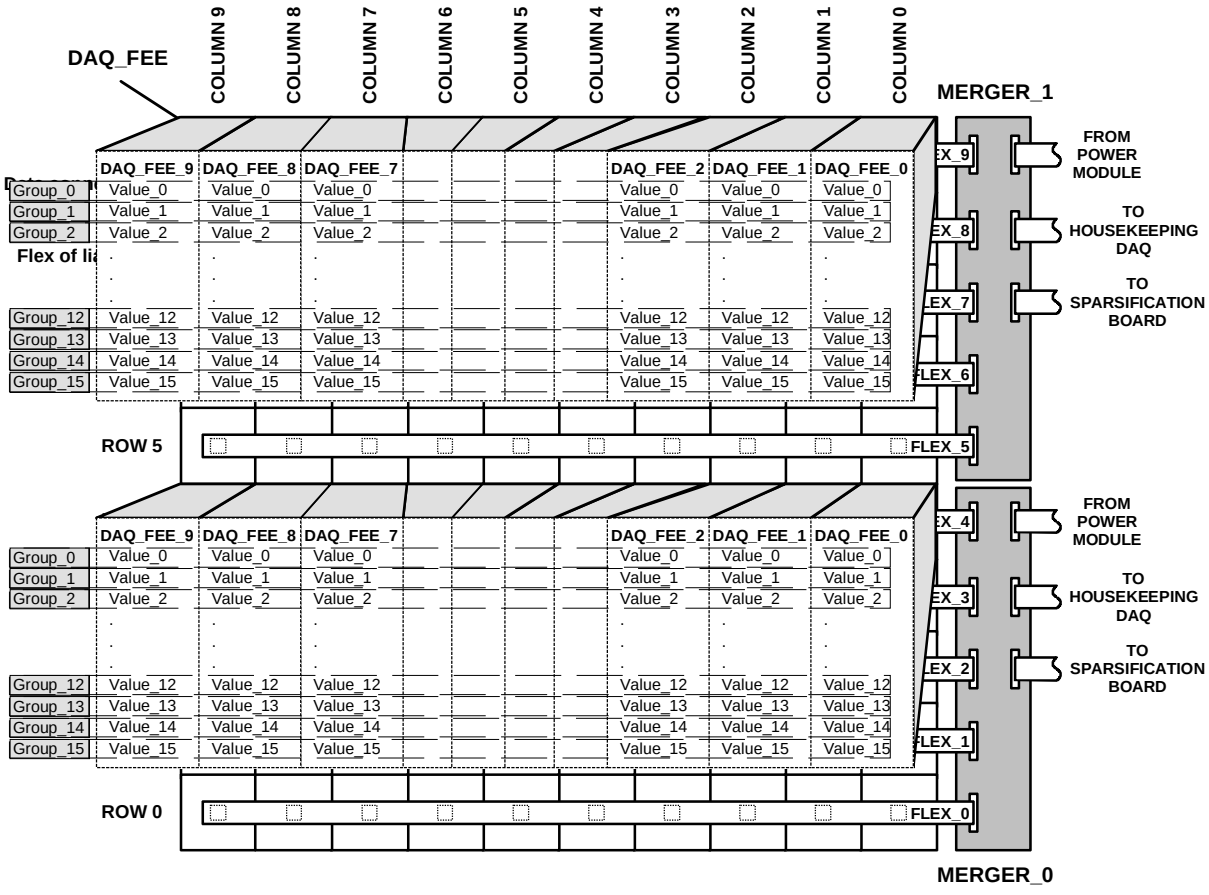


Figure 4 : Group words definition

VALUE_X is defined like the following description :

D[17…D14]	D13	D12	D[11…0]
Position Pmt	Fault	Gain	Pmt ADC values

4.2. MERGER board acquisition time

All the following calculations are done with a basis frequency equal to 40 MHz.

4.2.1. Row time acquisition calculation

The row time acquisition, from the DAQ_FEE board, is given by the following formula:

$$Trta = Tcst + Tmct + Tact + Tdfttd + Tfsmf$$ [3]

With

Trta : Row_time_acquisition : acquisition time for one row

Tcst : Clock_synchronisation_time : time where the command is passing through the MERGER FPGA . This value includes the time where the basis clock is valid before the generation of the command. This time is a multiple of the period_basis_clock
= 18 x 25 ns = 450 ns

Tmct : merger_command_time : command sending time from the MERGER board to the DAQ_FEE boards
= number_of_command_bit x number_of_basis_clock x period_basis_clock
= 8 x 1 x 25 ns = 200 ns

Tact : acknolewdge_command_time : time which permit an command interpretation by the DAQ_FEE board
= constant = 100 ns

Tdfttd : daq_fee_time_transmit_data : time necessary to transmit all the data through the DAQ_FEE boards to the MERGER board.
= (Acquisition time for one word) x Number of words
= (18 x 8 x 25 ns) x 16
= 3 600 x 16
= 57 600 ns

Tfsmf : Finite_State_machine_finished : Time required for finish a row data acquisition (a necessary time for the Row acquisition Finite State Machine to go in standby state)
= 8 x 25 ns
= 200 ns.

Trta = 450 ns + 200 ns + 100 ns + 57 600 ns + 200 ns = 58 550 ns = **58,55 µs**

4.2.2. Matrix time acquisition calculation

The matrix is managed by two MERGER board. Each of them acquire the data of half of the matrix. The two MERGER board working in parallel the entire matrix acquisition time is equal to the half matrix acquisition time made by one MERGER board.
The matrix time acquisition is equal to the row time acquisition multiplied by the number of rows managed by one MERGER board.

Tmta = Trta * 5 = 58,550 µs * 5 = **292,75 µs**

[3] Annex 6 : Row Data Acquisition Sequence shows graphically how the calculation is done.

5. Data acquisition transferring

After receiving the data from the DAQ_FEE board, the MERGER board must send these one to the SPARSIFICATION board through a high speed data link in a limited time to minimize the acquisition dead time.

The MERGER board must also add some information to the received data to identify precisely the data origin (which DAQ_FEE board have generated which data); This information are defined as the coordinates of the DAQ_FEE board in the matrix.
These coordinates are defined like below :

	COLUMN 9	COLUMN 8	COLUMN 7	COLUMN 6	COLUMN 5	COLUMN 4	COLUMN 3	COLUMN 2	COLUMN 1	COLUMN 0	
ROW 9	C: 1100 R: 1100	C: 1100 R: 1011	C: 1100 R: 1010	C: 1100 R: 1001	C: 1100 R: 1000	C: 1100 R: 0100	C: 1100 R: 0011	C: 1100 R: 0010	C: 1100 R: 0001	C: 1100 R: 0000	MERGER_1
ROW 8	C: 1011 R: 1100	C: 1011 R: 1011	C: 1011 R: 1010	C: 1011 R: 1001	C: 1011 R: 1000	C: 1011 R: 0100	C: 1011 R: 0011	C: 1011 R: 0010	C: 1011 R: 0001	C: 1011 R: 0000	
ROW 7	C: 1010 R: 1100	C: 1010 R: 1011	C: 1010 R: 1010	C: 1010 R: 1001	C: 1010 R: 1000	C: 1010 R: 0100	C: 1010 R: 0011	C: 1010 R: 0010	C: 1010 R: 0001	C: 1010 R: 0000	
ROW 6	C: 1001 R: 1100	C: 1001 R: 1011	C: 1001 R: 1010	C: 1001 R: 1001	C: 1001 R: 1000	C: 1001 R: 0100	C: 1001 R: 0011	C: 1001 R: 0010	C: 1001 R: 0001	C: 1001 R: 0000	
ROW 5	C: 1000 R: 1100	C: 1000 R: 1011	C: 1000 R: 1010	C: 1000 R: 1001	C: 1000 R: 1000	C: 1000 R: 0100	C: 1000 R: 0011	C: 1000 R: 0010	C: 1000 R: 0001	C: 1000 R: 0000	
ROW 4	C: 0100 R: 1100	C: 0100 R: 1011	C: 0100 R: 1010	C: 0100 R: 1001	C: 0100 R: 1000	C: 0100 R: 0100	C: 0100 R: 0011	C: 0100 R: 0010	C: 0100 R: 0001	C: 0100 R: 0000	
ROW 3	C: 0011 R: 1100	C: 0011 R: 1011	C: 0011 R: 1010	C: 0011 R: 1001	C: 0011 R: 1000	C: 0011 R: 0100	C: 0011 R: 0011	C: 0011 R: 0010	C: 0011 R: 0001	C: 0011 R: 0000	
ROW 2	C: 0010 R: 1100	C: 0010 R: 1011	C: 0010 R: 1010	C: 0010 R: 1001	C: 0010 R: 1000	C: 0010 R: 0100	C: 0010 R: 0011	C: 0010 R: 0010	C: 0010 R: 0001	C: 0010 R: 0000	
ROW 1	C: 0001 R: 1100	C: 0001 R: 1011	C: 0001 R: 1010	C: 0001 R: 1001	C: 0001 R: 1000	C: 0001 R: 0100	C: 0001 R: 0011	C: 0001 R: 0010	C: 0001 R: 0001	C: 0001 R: 0000	
ROW 0	C: 0000 R: 1100	C: 0000 R: 1011	C: 0000 R: 1010	C: 0000 R: 1001	C: 0000 R: 1000	C: 0000 R: 0100	C: 0000 R: 0011	C: 0000 R: 0010	C: 0000 R: 0001	C: 0000 R: 0000	MERGER_0

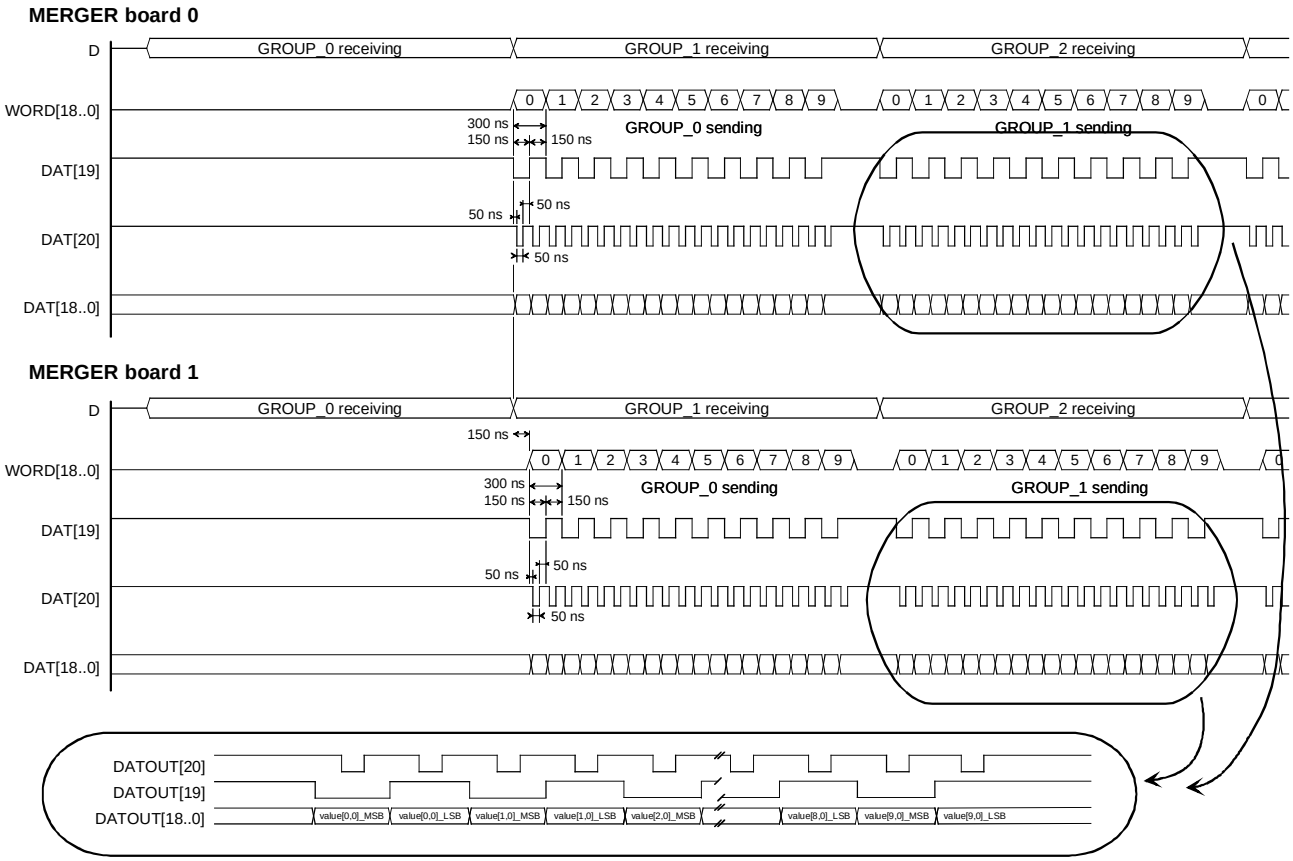
PMT MATRIX BOTTOM VIEW

The value are in binary representation, C represents Column and R represents Row.
These information can be add because the MERGER board knows on which rows and on which dedicated line it received each data.

To minimize the dead time and the MERGER board resources, the transferring principle is based on the fact that a data is transferred when it was received completely by chart MERGER, the transfer being made while the following data is to be received.
The MERGER board, receiving 10 data simultaneously (one data by line of DAQ_FEE boards) applies a priority on this one and transmits in first the data of the line 0 then that of line 1 and so on.
The physical interface between MERGER and SPARSIFICATION board is made by a couple of integrated circuit : one of them is a parallel to serial transmitter (located on the MERGER board) and the other is a serial to parallel receiver (located on the SPARSIFICATION board). These circuits are designed for a high speed serial link.
These circuits works with a 21 bit parallel bus, more than the one word to be transmit (26 bits). The transmission of data thus requires two transfers in accordance with the signal waveform locates in the

following chapter (chapter 5.1 Output signal waveform with a 21 bits Channel link page 9).
the description of the words and the transmitted bits is made in a chapter which follows (chapter 5.2 Output word assigning bits with a 21 bits Channel link page 9).

5.1. Output signal waveform with a 21 bits Channel link



5.2. Output word assigning bits with a 21 bits Channel link

The MERGER board sends all the data's in two 21 bits words. The MERGER board sends first WORD0 then WORD1. This 2 words are defined as below :

WORD0								
Label	/Valid DATA	Low/High	Unused	Fault	Gain Value	Column Number	Row Number	Channel Number
Output	D[20]	D[19]	D[18..14]	D[13]	D[12]	D[11..8]	D[7..4]	D[3..0]
Data	'Valid DATA'	'0'	"00000"	'Fault'	'Gain Value'	'Column Number'	'Row Number'	'Channel Number'
Number of bits	1	1	5	1	1	4	4	4

WORD1				
Label	/Valid DATA	Low/High	Unused	ADC DATA
Output	D[20]	D[19]	D[18..12]	D[11..0]
Data	'Valid DATA'	'1'	"0000000"	"ADC_DATA"
Number of bits	1	1	7	12

Common data word signification :

D[21] Valid_DATA bit which indicate when set to 1 that the word is a valid data
D[20] Low/High Indicate when set to :
0 the word is the Most Significant Word of the acquired data
1 the word is the Low Significant Word of the acquired data and the last word of the data

Word 0 Specific bit signification :

D[13] Fault when set to '1' indicate that the associated module wire is malfunctioning. Normally this bit is '0' state.
D[12] Gain Value indicates which gain is associated to the ADC data value :
- ✓ '0' : gain 5
- ✓ '1' : gain 1

D[11..8] Column Number Indicates on which column the data has been taken. The defined value are 0 to 4 and 8 to C (in hexadecimal notation)
D[7..4] Row number Indicates on which row the data has been taken. The defined value are 0 to 4 and 8 to C (in hexadecimal notation)
D[3..0] Channel Number Indicates the number of sensor in the DAQ_FEE board which the ADC_DATA is associated.

Word 1 Specific bit signification :

D[11..0] ADC DATA Digital channel value

All the unused bits are send as low level logic.
For more information about the bit definition see the DAQ_FEE board documentation.

5.3. MERGER board sending time

Here is the calculation of the time necessary for transmitting an acquired data row to the SPARSIFICATION board.
To have a constant flow of dated from the DAQ_FEE boards to the SPARSIFICATION board, this time must be lower than the Row time acquisition (Tdata_group) calculated above.
The sending time, to SPARSIFICATION board, between 2 data's must be at least equal to **300 ns** (for more information about this timing see the SPARSIFICATION board documentation)
The data group, which must be transferred, contains 10 words.

The group sending time can be calculating according to the formula :

Tsending_time = number_of_word x time_between_2_data = 10 x 300 ns = **3 µs**

6. Matrix acquisition time calculation

The matrix acquisition time can be expresses by the sum of :

- TRIGGER command time duration
- TRIGGER command MEGER board internal delay
- Analog to Digital time duration for all the channel
- Matrix reading acquisition time

6.1. TRIGGER command time duration

The TRIGGER command is a command without argument and it's defined like a 8 bit command.
Its duration is equal to 8 x basis_clock = 8 x 25 ns = 200 ns

6.2. TRIGGER command MERGER board internal delay

The internal delay is equal to 10 basis_clock (250 ns = 10 x 25 ns).

6.3. Analog to Digital conversion time duration for all the channel

This time is defined like a constant and it's equal to 47,6 µs.

6.4. Matrix reading acquisition time

With the principle of reading/transmitting explains in a previous chapter, the matrix reading acquisition time can be expresses as follows

Matrix_reading_acquisition_time = Row_reading_acquisition_time * number_of_row + last_row_sending_time
= 58,55 µs * 5 + 3 µs

Matrix_reading_acquisition_time = 295,75 µs

Matrix Acquisition time = Trigger command time duration
+ trigger command MERGER board internal delay
+ Analog to Digital time duration
+ Matrix reading acquisition time
= 200 ns + 250 ns + 47 600 ns + 295 750 ns
= 342 600 ns

Matrix Acquisition time = 343,8 µs

7. FPGA "MERGER"

The design of this board is based on a FPGA (ACTEL PROASIC PLUS) and driver/receiver components.
All the complete functions are defined in the FPGA which are the core of this board.

7.1. Internal architecture

Rule : Each signal output is driven by a flip flop for having a constant delay between this output and the basis clock. The calculate delay is easiest to know.

Rule : In the most case each input is sample by one Flip flop for having a good functionally of the Finite State machine.

The design is divided in 3 subsystem :

- ✓ Command module
- ✓ Core module
- ✓ IO_data module
- ✓ DAC module

The Command module manages the command reception (from the SPARSIFICATION box) and the command emitting to the DAQ_FEE boards. This is the module which generates the Enable To Transmit Data command to the selected row.
The Core module is the primary Finite State Machine (FSM) which manages the good working of the “MERGER” FPGA.
The IO_data module manages the storage of the data send by the DAQ_FEE boards and the sending of them to the SPARSIFICATION box.
The DAC module manage the interface between the FPGA and the Digital to Analog converter component which define the analog power applied to the calibration LED.

A synoptic concerning the design internal general architecture is shown in Annex 7 page 27.

7.1.1. Core module architecture

A MERGER CORE sub-module architecture is shown in Annex 8 : MAIN and SLAVE MERGER_CORE sub-module architecture page 28.

7.1.2. IO_data module architecture

An IO data module architecture is shown in Annex 9 : MERGER_DATA_IO sub-module architecture page 31. This annex is divided in different figure which represent the different element which is inside.

7.2. Package

7.2.1. Pin description

SPARFICATION Box connection :

Ckin :	basis clock,	generated by the SPARFICATION board, nominal value : 40MHz
RSTin_N :	RESET signal,	low level active. This signal is coming back the SPARSIFICATION board.
Din :	DATA in.	This serial data link coming from the SPARSIFICATION board receives the acquisition command.
Enin_N :	/ENABLE in,	low level active. This serial link coming from the SPARSIFICATION board is the enable signal for the DATA_in line.
PD_N :	/Power_Down	Output. This signal controls the selection of the SERDES component. When set to 0, put the SERDES component in low power but, in this application, always set to 1.
CK_SERDES :	Transmit clock	Output. Generated by the FPGA to the SERDES component for the data transmission through the SERDES components to the SPARSIFICATION board. Nominal value : 20 MHz. This signal is generated only with a 21 bits SERDES link. With a 28 bits SERDES link this function is assumed by the basis clock, Ckin.

DAT[20..0] : DATA_OUT[20..0] Output, data bus to the SPARSIFICATION board. The number of bits is function of the SERDES type link :

Figure 10 21 bits (DATA_OUT(20..0)) for a 21 SERDES link

Internal board connection :

NO_ME : board number. This signal is set by a resistor jumper to low or high level
Low level : MERGER board number 0
High level : MERGER board number 1

POR_RESET : Power on Reset : high level active, this signal become active at the power on of the MERGER board, It is generated by the 5V/3V3 DC/DC converter included on the board.

Rows connection :

RST[4..0] :	ROW RESET,	output, high level active, Reset signal for the DAQ_FEE board placed on the row. Each row has his own signal.
ROW :	ROW[4..0][3],	output. More significant bit of the address of the row. It is only the replicate state of the NO_ME input. Each row has its own signal
Dout :	Data out[4..0].	output, Serial data link to the DAQ_FEE board placed on the row (the command is sent on this line. Each row has its own signal
EN_N :	ENABLE[4..0],	output, low level active. This signal is the enable of the data which sent on the Data out line. Each row has its own signal
SEL :	SELECT[4..0],	output, high level active. Selects the output driver when the ROW is selected to receive a command. Each row has its own signal.
ENDR :	ENDR[4..0],	Enable Data to Receive, output, high level active. Selects the input receiver when the ROW is selected to transmit a data. Each row has its own signal.
DR[9..0] :	DR[4..0][9..0],	Data Return. Signal data from the DAQ_FEE board. There are 10 data signal by row, each coming from each DAQ_FEE board placed on the ROW. Each row has its own signals.
PW_ROW	PW_ROW[4..0]	Row Power Enable : high level active. When this bit is set to one the DAQ_FEE present on the row is powered.

LED connection :

LED_VAL :	LED_VAL,	output, high level active, Validation signal for the LED. This one is powered only when this signal is active.
LED_DAC_D :	LED_DAC_D,	output. Serial data signal for the Digital to Analog converter.
LED_DAC_CK :	LED_DAC_CK,	output, rising edge sensitive. Serial Clock signal for the Digital to Analog converter.
LED_DAC_EN :	LED_DAC_EN,	output, low level active. Enable signal for the Digital to Analog Converter. This component is selected only if this signal is active.
SPARE :	SPARE,	spare output for future applications.

7.2.2. Pin count

The design needs 119 Input/Output pins distributed in the following way :

- ✓ 56 inputs
- ✓ 63 outputs

7.2.2.1. Connection with the SPARSIFICATION box

The connection with the SPARSIFICATION board use 4 inputs and 23 outputs defined like below :

- ✓ Ckin : input
- ✓ Din : input
- ✓ Enin_N : input
- ✓ RSTin_N : input
- ✓ PD_N : output
- ✓ CK_SERDES : output
- ✓ DAT[20..0] : output

7.2.2.2. Connection with the rows

The following counting must be multiplied by 5 because the board drives 5 rows.
The connection, with one row, use 10 inputs and 7 Outputs defined like below :

- ✓ RST : output
- ✓ ROW : output
- ✓ Dout : output
- ✓ EN_N : output
- ✓ SEL : output
- ✓ ENDR : output
- ✓ DR[9..0] : 10 inputs
- ✓ PW_ROW : output

7.2.2.3. Connection with the LED

- ✓ LED_VAL : output
- ✓ LED_DAC_D : output
- ✓ LED_DAC_CK : output
- ✓ LED_DAC_EN : output
- ✓ SPARE : output

7.2.2.4. Flag input

Input/Output connected to a signal used only on the board (2 inputs)

- ✓ NO_ME : input
- ✓ POR_RESET : input

7.2.3. Component pin out

7.2.3.1. Component pin out (listed by name)

Pin Report – 2006 Pinchecksum: NOT-AVAILABLE
Product: Designer
Release: v7.0
Version: 7.0.0.11

Design Name: merger Family: PA Package: 456 BGA

Port	Pin	Function	Output Load
CK_SERDES	K25		35
CLK	M1	GL1	---
DAC_DIN	M23		35
DAC_SCLK	P22		35
DAC_SYNC_N	P23		35
DAT[0]	L26		35
DAT[1]	L25		35
DAT[2]	M26		35
DAT[3]	M25		35
DAT[4]	P25		35
DAT[5]	R26		35
DAT[6]	R25		35
DAT[7]	T26		35
DAT[8]	T25		35
DAT[9]	U26		35
DAT[10]	U25		35
DAT[11]	V26		35
DAT[12]	V25		35
DAT[13]	W26		35
DAT[14]	W25		35
DAT[15]	Y26		35
DAT[16]	Y25		35
DAT[17]	AA26		35
DAT[18]	AA25		35
DAT[19]	AB25		35
DAT[20]	K26		35
DAT_IN_ACK	AB1		35
DIN	AB10		---
dout_row[0]	P3		35
dout_row[1]	G3		35
dout_row[2]	B13		35
dout_row[3]	C17		35
dout_row[4]	G24		35
dr_row0[0]	R2		---
dr_row0[1]	R1		---
dr_row0[2]	T2		---
dr_row0[3]	T1		---
dr_row0[4]	U2		---
dr_row0[5]	U1		---
dr_row0[6]	V2		---
dr_row0[7]	V1		---
dr_row0[8]	W2		---
dr_row0[9]	W1		---
dr_row1[0]	E2		---
dr_row1[1]	E1		---
dr_row1[2]	F2		---
dr_row1[3]	F1		---

Port	Pin	Function	Output Load
dr_row1[4]	G2		---
dr_row1[5]	G1		---
dr_row1[6]	H2		---
dr_row1[7]	H1		---
dr_row1[8]	J2		---
dr_row1[9]	J1		---
dr_row2[0]	A13		---
dr_row2[1]	B12		---
dr_row2[2]	A12		---
dr_row2[3]	A11		---
dr_row2[4]	B11		---
dr_row2[5]	A10		---
dr_row2[6]	B10		---
dr_row2[7]	A9		---
dr_row2[8]	B9		---
dr_row2[9]	A8		---
dr_row3[0]	B19		---
dr_row3[1]	A19		---
dr_row3[2]	B18		---
dr_row3[3]	A18		---
dr_row3[4]	B17		---
dr_row3[5]	A17		---
dr_row3[6]	B16		---
dr_row3[7]	A16		---
dr_row3[8]	B15		---
dr_row3[9]	A15		---
dr_row4[0]	J26		---
dr_row4[1]	J25		---
dr_row4[2]	H26		---
dr_row4[3]	H25		---
dr_row4[4]	G26		---
dr_row4[5]	G25		---
dr_row4[6]	F26		---
dr_row4[7]	F25		---
dr_row4[8]	E26		---
dr_row4[9]	E25		---
en_n_row[0]	R3		35
en_n_row[1]	E3		35
en_n_row[2]	A14		35
en_n_row[3]	C19		35
en_n_row[4]	H24		35
endr_row[0]	V3		35
endr_row[1]	H3		35
endr_row[2]	C10		35
endr_row[3]	C16		35
endr_row[4]	F24		35
ENIN_N	AB9		---

Port	Pin	Function	Output Load
LED_CMD	R24		35
LED_DAC_CK	P22		35
LED_DAC_D	M23		35
LED_DAC_EN	P23		35
NO_ME	AC17		---
PD_N	K24		35
por_reset_g	M22	GL4	---
pw_row[0]	Y2		35
pw_row[1]	D3		35
pw_row[2]	C13		35
pw_row[3]	C15		35
pw_row[4]	E24		35
row3_row[0]	T3		35
row3_row[1]	C2		35
row3_row[2]	B14		35

Port	Pin	Function	Output Load
row3_row[3]	C20		35
row3_row[4]	J24		35
rst_row[0]	Y1		35
rst_row[1]	F3		35
rst_row[2]	C12		35
rst_row[3]	C14		35
rst_row[4]	D24		35
rstin_n_g	N23	GL3	---
sel_row[0]	U3		35
sel_row[1]	H4		35
sel_row[2]	C11		35
sel_row[3]	C18		35
sel_row[4]	J23		35
SPARE	M24		35
trig_received	AB16		35

7.2.3.2. Component pin out (listed by pin)

Pin Report 2006 Pin checksum: NOT-AVAILABLE
Product: Designer
Release: v7.0
Version: 7.0.0.11
Design Name: merger Family: PA Package: 456 BGA

Nber	Port	Function	State
A1		VDDP	Reserved
A2		VDDP	Reserved
A3		Not Bonded	Unconnected
A4		Not Bonded	Unconnected
A5		Not Bonded	Unconnected
A6		Not Bonded	Unconnected
A7		Not Bonded	Unconnected
A8	dr_row2[9]		Fixed
A9	dr_row2[7]		Fixed
A10	dr_row2[5]		Fixed
A11	dr_row2[3]		Fixed
A12	dr_row2[2]		Fixed
A13	dr_row2[0]		Fixed
A14	en_n_row[2]		Fixed
A15	dr_row3[9]		Fixed
A16	dr_row3[7]		Fixed
A17	dr_row3[5]		Fixed
A18	dr_row3[3]		Fixed
A19	dr_row3[1]		Fixed
A20		Not Bonded	Unconnected
A21		Not Bonded	Unconnected
A22		Not Bonded	Unconnected
A23		Not Bonded	Unconnected
A24		Not Bonded	Unconnected

Nber	Port	Function	State
A25		VDDP	Reserved
A26		VDDP	Reserved
AA1			Unassigned
AA2			Unassigned
AA3			Unassigned
AA4			Unassigned
AA5		VDD	Reserved
AA22		VDD	Reserved
AA23			Unassigned
AA24			Unassigned
AA25	DAT[18]		Fixed
AA26	DAT[17]		Fixed
AB1	DAT_IN_ACK		Fixed
AB2			Unassigned
AB3			Unassigned
AB4			Unassigned
AB5		VDD	Reserved
AB6		VDD	Reserved
AB7		VDD	Reserved
AB8			Unassigned
AB9	ENIN_N		Fixed
AB10	DIN		Fixed
AB11			Unassigned
AB12			Unassigned

Nber	Port	Function	State
AB13			Unassigned
AB14			Unassigned
AB15			Unassigned
AB16	TRIG_RECEIVED		Fixed
AB17			Unassigned
AB18			Unassigned
AB19			Unassigned
AB20		VDD	Reserved
AB21		VDD	Reserved
AB22		VDD	Reserved
AB23			Unassigned
AB24			Unassigned
AB25	DAT[19]		Fixed
AB26		Not Bonded	Unconnected
AC1			Unassigned
AC2			Unassigned
AC3			Unassigned
AC4		VDDP	Reserved
AC5		Not Bonded	Unconnected
AC6			Unassigned
AC7			Unassigned
AC8			Unassigned
AC9			Unassigned
AC10			Unassigned
AC11			Unassigned
AC12			Unassigned
AC13			Unassigned
AC14			Unassigned
AC15			Unassigned
AC16			Unassigned
AC17	NO_ME		Fixed
AC18			Unassigned
AC19			Unassigned
AC20			Unassigned
AC21		TMS	Reserved
AC22		TDO	Reserved
AC23		VDDP	Reserved
AC24		RCK	Reserved
AC25		Not Bonded	Unconnected
AC26			Unassigned
AD1		Not Bonded	Unconnected
AD2			Unassigned
AD3		VDDP	Reserved
AD4		Not Bonded	Unconnected
AD5		Not Bonded	Unconnected
AD6		Not Bonded	Unconnected
AD7			Unassigned
AD8			Unassigned
AD9			Unassigned

Nber	Port	Function	State
AD10			Unassigned
AD11			Unassigned
AD12			Unassigned
AD13			Unassigned
AD14			Unassigned
AD15			Unassigned
AD16			Unassigned
AD17			Unassigned
AD18			Unassigned
AD19			Unassigned
AD20		Not Bonded	Unconnected
AD21		TCK	Reserved
AD22		VPP	Reserved
AD23		Not Bonded	Unconnected
AD24		VDDP	Reserved
AD25		Not Bonded	Unconnected
AD26		Not Bonded	Unconnected
AE1		VDDP	Reserved
AE2		VDDP	Reserved
AE3		Not Bonded	Unconnected
AE4		Not Bonded	Unconnected
AE5		Not Bonded	Unconnected
AE6		Not Bonded	Unconnected
AE7		Not Bonded	Unconnected
AE8			Unassigned
AE9			Unassigned
AE10			Unassigned
AE11			Unassigned
AE12			Unassigned
AE13			Unassigned
AE14			Unassigned
AE15			Unassigned
AE16			Unassigned
AE17			Unassigned
AE18			Unassigned
AE19			Unassigned
AE20		Not Bonded	Unconnected
AE21		Not Bonded	Unconnected
AE22		Not Bonded	Unconnected
AE23		VPN	Reserved
AE24		TRST	Reserved
AE25		VDDP	Reserved
AE26		VDDP	Reserved
AF1		VDDP	Reserved
AF2		VDDP	Reserved
AF3		Not Bonded	Unconnected
AF4		Not Bonded	Unconnected
AF5		Not Bonded	Unconnected
AF6		Not Bonded	Unconnected

Nber	Port	Function	State
AF7		Not Bonded	Unconnected
AF8		Not Bonded	Unconnected
AF9			Unassigned
AF10			Unassigned
AF11			Unassigned
AF12			Unassigned
AF13			Unassigned
AF14			Unassigned
AF15			Unassigned
AF16			Unassigned
AF17			Unassigned
AF18		Not Bonded	Unconnected
AF19		Not Bonded	Unconnected
AF20		Not Bonded	Unconnected
AF21		Not Bonded	Unconnected
AF22		Not Bonded	Unconnected
AF23		TDI	Reserved
AF24		Not Bonded	Unconnected
AF25		VDDP	Reserved
AF26		VDDP	Reserved
B1		VDDP	Reserved
B2		VDDP	Reserved
B3		Not Bonded	Unconnected
B4		Not Bonded	Unconnected
B5		Not Bonded	Unconnected
B6		Not Bonded	Unconnected
B7		Not Bonded	Unconnected
B8			Unassigned
B9	dr_row2[8]		Fixed
B10	dr_row2[6]		Fixed
B11	dr_row2[4]		Fixed
B12	dr_row2[1]		Fixed
B13	dout_row[2]		Fixed
B14	row3_row[2]		Fixed
B15	dr_row3[8]		Fixed
B16	dr_row3[6]		Fixed
B17	dr_row3[4]		Fixed
B18	dr_row3[2]		Fixed
B19	dr_row3[0]		Fixed
B20		Not Bonded	Unconnected
B21		Not Bonded	Unconnected
B22		Not Bonded	Unconnected
B23		Not Bonded	Unconnected
B24		Not Bonded	Unconnected
B25		VDDP	Reserved
B26		VDDP	Reserved
C1		VDDP	Reserved
C2	row3_row[1]		Fixed
C3		VDDP	Reserved

Nber	Port	Function	State
C4		Not Bonded	Unconnected
C5		Not Bonded	Unconnected
C6		Not Bonded	Unconnected
C7			Unassigned
C8			Unassigned
C9			Unassigned
C10	endr_row[2]		Fixed
C11	sel_row[2]		Fixed
C12	rst_row[2]		Fixed
C13	pw_row[2]		Fixed
C14	rst_row[3]		Fixed
C15	pw_row[3]		Fixed
C16	endr_row[3]		Fixed
C17	dout_row[3]		Fixed
C18	sel_row[3]		Fixed
C19	en_n_row[3]		Fixed
C20	row3_row[3]		Fixed
C21		Not Bonded	Unconnected
C22		Not Bonded	Unconnected
C23		Not Bonded	Unconnected
C24		VDDP	Reserved
C25		Not Bonded	Unconnected
C26		Not Bonded	Unconnected
D1		Not Bonded	Unconnected
D2		Not Bonded	Unconnected
D3	pw_row[1]		Fixed
D4		VDDP	Reserved
D5		Not Bonded	Unconnected
D6		Not Bonded	Unconnected
D7			Unassigned
D8			Unassigned
D9			Unassigned
D10			Unassigned
D11			Unassigned
D12			Unassigned
D13			Unassigned
D14			Unassigned
D15			Unassigned
D16			Unassigned
D17			Unassigned
D18			Unassigned
D19			Unassigned
D20			Unassigned
D21			Unassigned
D22		Not Bonded	Unconnected
D23		VDDP	Reserved
D24	rst_row[4]		Fixed
D25		Not Bonded	Unconnected
D26		Not Bonded	Unconnected

Nber	Port	Function	State
E1	dr_row1[1]		Fixed
E2	dr_row1[0]		Fixed
E3	en_n_row[1]		Fixed
E4			Unassigned
E5		VDD	Reserved
E6		VDD	Reserved
E7		VDD	Reserved
E8		VDD	Reserved
E9			Unassigned
E10			Unassigned
E11			Unassigned
E12			Unassigned
E13			Unassigned
E14			Unassigned
E15			Unassigned
E16			Unassigned
E17			Unassigned
E18			Unassigned
E19			Unassigned
E20		VDD	Reserved
E21		VDD	Reserved
E22		VDD	Reserved
E23			Unassigned
E24	pw_row[4]		Fixed
E25	dr_row4[9]		Fixed
E26	dr_row4[8]		Fixed
F1	dr_row1[3]		Fixed
F2	dr_row1[2]		Fixed
F3	rst_row[1]		Fixed
F4			Unassigned
F5		VDD	Reserved
F22		VDD	Reserved
F23			Unassigned
F24	endr_row[4]		Fixed
F25	dr_row4[7]		Fixed
F26	dr_row4[6]		Fixed
G1	dr_row1[5]		Fixed
G2	dr_row1[4]		Fixed
G3	dout_row[1]		Fixed
G4			Unassigned
G5		VDD	Reserved
G22		VDD	Reserved
G23			Unassigned
G24	dout_row[4]		Fixed
G25	dr_row4[5]		Fixed
G26	dr_row4[4]		Fixed
H1	dr_row1[7]		Fixed
H2	dr_row1[6]		Fixed
H3	endr_row[1]		Fixed

Nber	Port	Function	State
H4	sel_row[1]		Fixed
H5		VDD	Reserved
H22		VDD	Reserved
H23			Unassigned
H24	en_n_row[4]		Fixed
H25	dr_row4[3]		Fixed
H26	dr_row4[2]		Fixed
J1	dr_row1[9]		Fixed
J2	dr_row1[8]		Fixed
J3			Unassigned
J4			Unassigned
J5			Unassigned
J22			Unassigned
J23	sel_row[4]		Fixed
J24	row3_row[4]		Fixed
J25	dr_row4[1]		Fixed
J26	dr_row4[0]		Fixed
K1			Unassigned
K2			Unassigned
K3			Unassigned
K4			Unassigned
K5			Unassigned
K22			Unassigned
K23			Unassigned
K24	PD_N		Fixed
K25	CK_SERDES		Fixed
K26	DAT[20]		Fixed
L1			Unassigned
L2			Unassigned
L3			Unassigned
L4			Unassigned
L5			Unassigned
L11		GND	Reserved
L12		GND	Reserved
L13		GND	Reserved
L14		GND	Reserved
L15		GND	Reserved
L16		GND	Reserved
L22			Unassigned
L23			Unassigned
L24			Unassigned
L25	DAT[1]		Fixed
L26	DAT[0]		Fixed
M1	CLK	GL1	Special
M2		GL2	Special
M3			Unassigned
M4			Unassigned
M5			Unassigned
M11		GND	Reserved

Nber	Port	Function	State
M12		GND	Reserved
M13		GND	Reserved
M14		GND	Reserved
M15		GND	Reserved
M16		GND	Reserved
M22	POR_RESET_G	GL4	Special
M23	DAC_DIN		Fixed
M24	SPARE		Fixed
M25	DAT[3]		Fixed
M26	DAT[2]		Fixed
N1			Unassigned
N2		I/O(GLMX1)	Unassigned
N3		AGND	Reserved
N4		PPECL1(I/P)	Special
N5		AVDD	Reserved
N11		GND	Reserved
N12		GND	Reserved
N13		GND	Reserved
N14		GND	Reserved
N15		GND	Reserved
N16		GND	Reserved
N22		NPECL2	Special
N23	RSTIN_N_G	GL3	Special
N24		AVDD	Reserved
N25		I/O(GLMX2)	Unassigned
N26		AGND	Reserved
P1			Unassigned
P2			Unassigned
P3	dout_row[0]		Fixed
P4			Unassigned
P5		NPECL1	Special
P11		GND	Reserved
P12		GND	Reserved
P13		GND	Reserved
P14		GND	Reserved
P15		GND	Reserved
P16		GND	Reserved
P22	DAC_CLK		Fixed
P23	DAC_SYNC_N		Fixed
P24			Unassigned
P25	DAT[4]		Fixed
P26		PPECL2(I/P)	Special
R1	dr_row0[1]		Fixed
R2	dr_row0[0]		Fixed
R3	en_n_row[0]		Fixed
R4			Unassigned
R5			Unassigned
R11		GND	Reserved
R12		GND	Reserved

Nber	Port	Function	State
R13		GND	Reserved
R14		GND	Reserved
R15		GND	Reserved
R16		GND	Reserved
R22			Unassigned
R23			Unassigned
R24	LED_CMD		Fixed
R25	DAT[6]		Fixed
R26	DAT[5]		Fixed
T1	dr_row0[3]		Fixed
T2	dr_row0[2]		Fixed
T3	row3_row[0]		Fixed
T4			Unassigned
T5			Unassigned
T11		GND	Reserved
T12		GND	Reserved
T13		GND	Reserved
T14		GND	Reserved
T15		GND	Reserved
T16		GND	Reserved
T22			Unassigned
T23			Unassigned
T24			Unassigned
T25	DAT[8]		Fixed
T26	DAT[7]		Fixed
U1	dr_row0[5]		Fixed
U2	dr_row0[4]		Fixed
U3	sel_row[0]		Fixed
U4			Unassigned
U5			Unassigned
U22			Unassigned
U23			Unassigned
U24			Unassigned
U25	DAT[10]		Fixed
U26	DAT[9]		Fixed
V1	dr_row0[7]		Fixed
V2	dr_row0[6]		Fixed
V3	endr_row[0]		Fixed
V4			Unassigned
V5			Unassigned
V22			Unassigned
V23			Unassigned
V24			Unassigned
V25	DAT[12]		Fixed
V26	DAT[11]		Fixed
W1	dr_row0[9]		Fixed
W2	dr_row0[8]		Fixed
W3			Unassigned
W4			Unassigned

Nber	Port	Function	State
W5		VDD	Reserved
W22		VDD	Reserved
W23			Unassigned
W24			Unassigned
W25	DAT[14]		Fixed
W26	DAT[13]		Fixed
Y1	rst_row[0]		Fixed
Y2	pw_row[0]		Fixed

Nber	Port	Function	State
Y3			Unassigned
Y4			Unassigned
Y5		VDD	Reserved
Y22		VDD	Reserved
Y23			Unassigned
Y24			Unassigned
Y25	DAT[16]		Fixed
Y26	DAT[15]		Fixed

7.2.4. FPGA consumption

Power Report for design merger with the following settings :

Vendor = Actel Corporation
Program = Actel Designer Software, Release v7.0, Copyright © 1989-2005
Date = Sat Feb 11 20:20:25 2006
Version = 01.01.01

Family = PA
Die = APA300
Package = 456 BGA
Temperature = IND
Voltage = IND
Speed = STD
Conditions = Typical

Exploration Max Depth = None
Exploration Min Ratio = None
Exploration Min Value = None
Report Static Power = Yes
Report Dynamic Power = Yes
Report Power Breakdown = No
Flattened Power Report = No

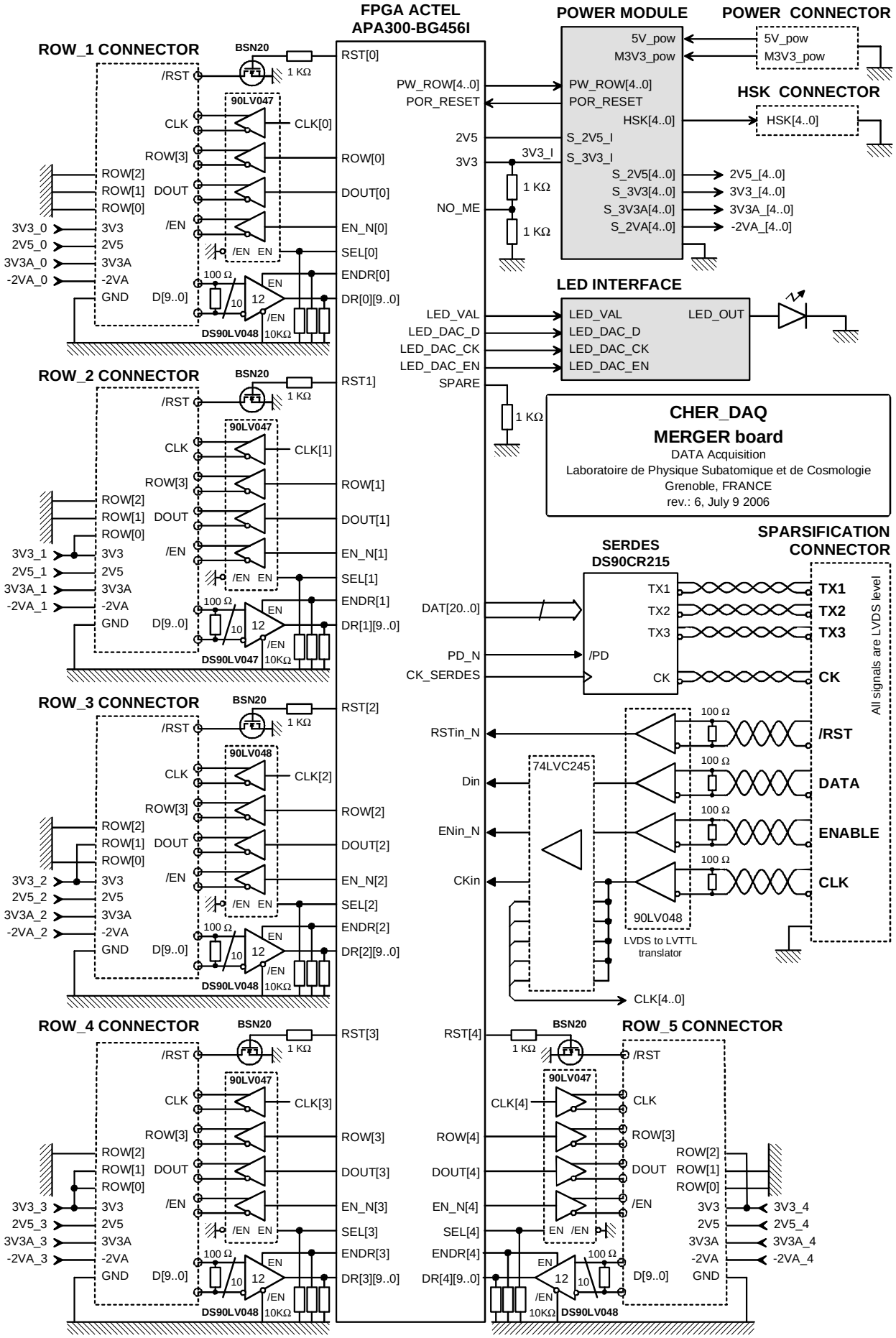
Section Static Power Summary
Block merger, 11.000000 mW
End Section

Section Dynamic Power Summary
Block merger, 223.727904 mW
End Section

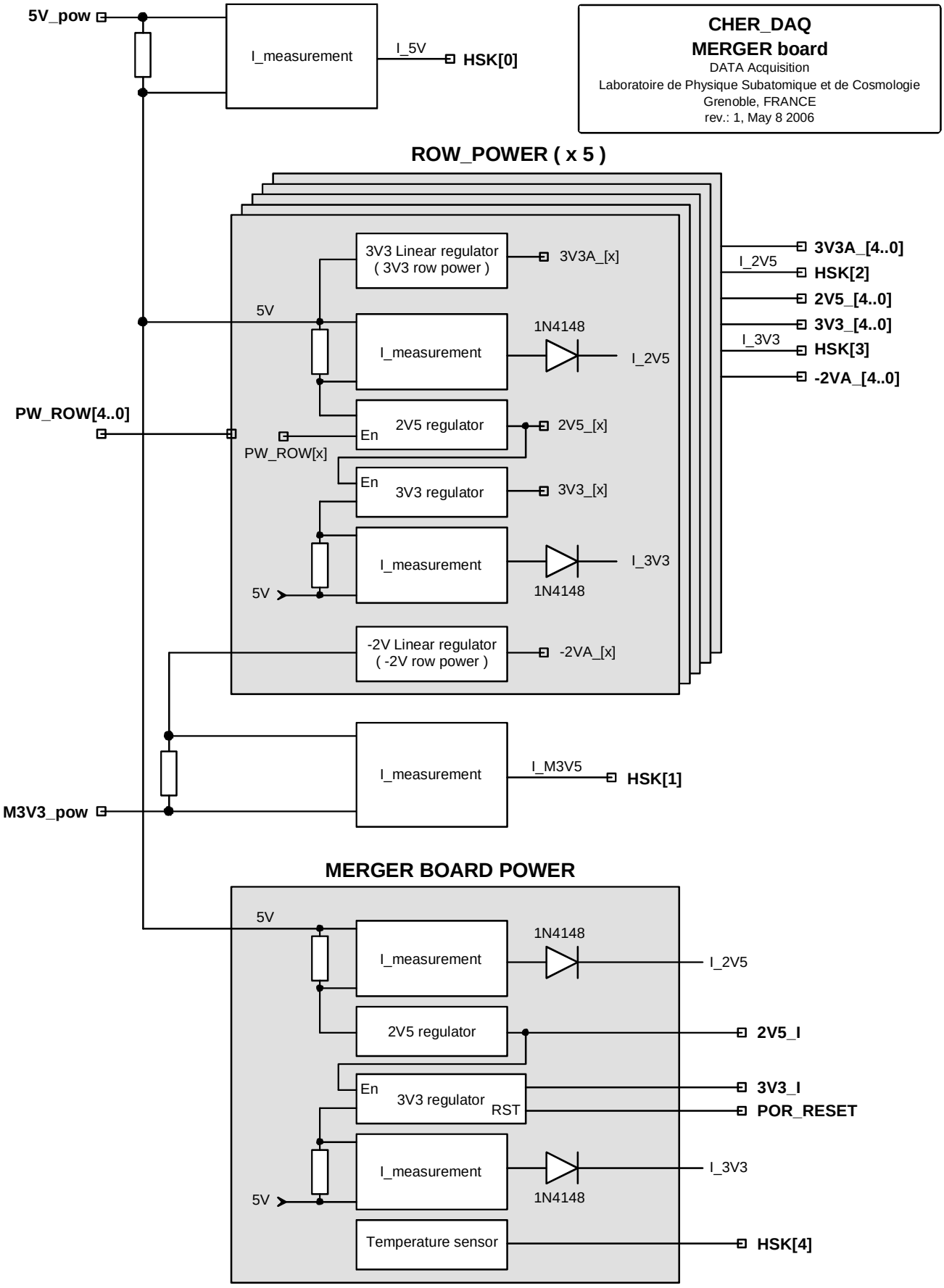
8. Reference documentation

- ✓ CHER_DAQ : interface document, O. BOURRION
- ✓ CHERCAM : SPARSIFICATION box, O. BOURRION
- ✓ DAQ_FEE, front end electronics for CHERCAM detector, B. BOYER
- ✓ MAX9173 : Quad LVDS Line Receiver with Flow-Through Pinout and “In-Path” Fail-Safe, MAXIM
- ✓ MAX9123 : Quad LVDS Line Driver with Flow-Through Pinout, MAXIM

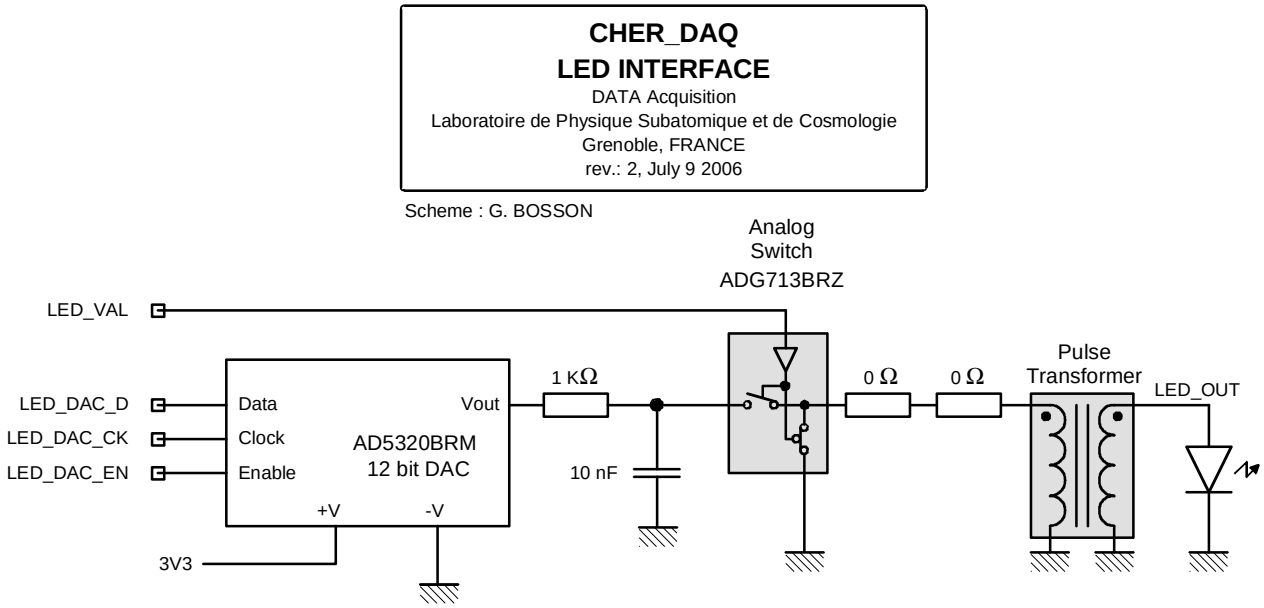
Annex 1 : Logical parts block diagram with 21 bits SERDES



Annex 2 : Power parts block diagram

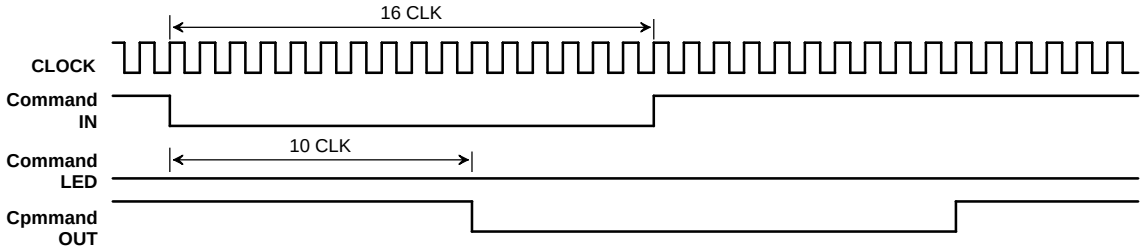


Annex 3 : LED interface diagram

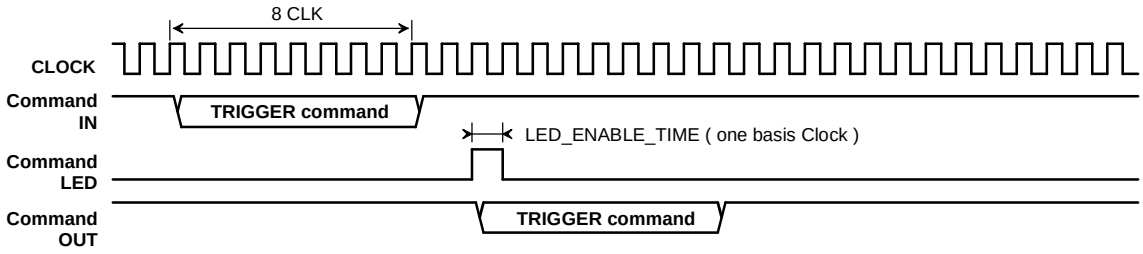


Annex 4 : LED command sequence

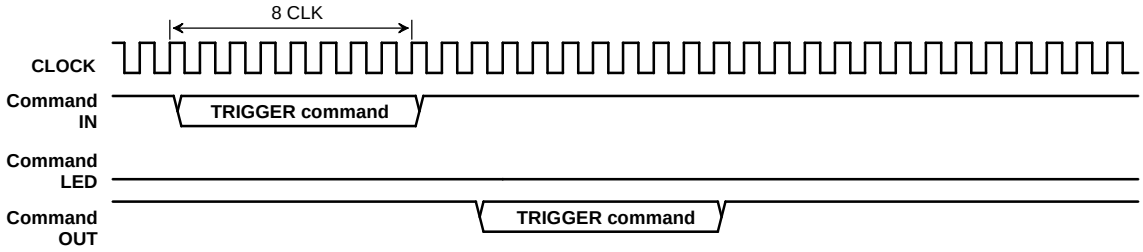
LED OFF/ON



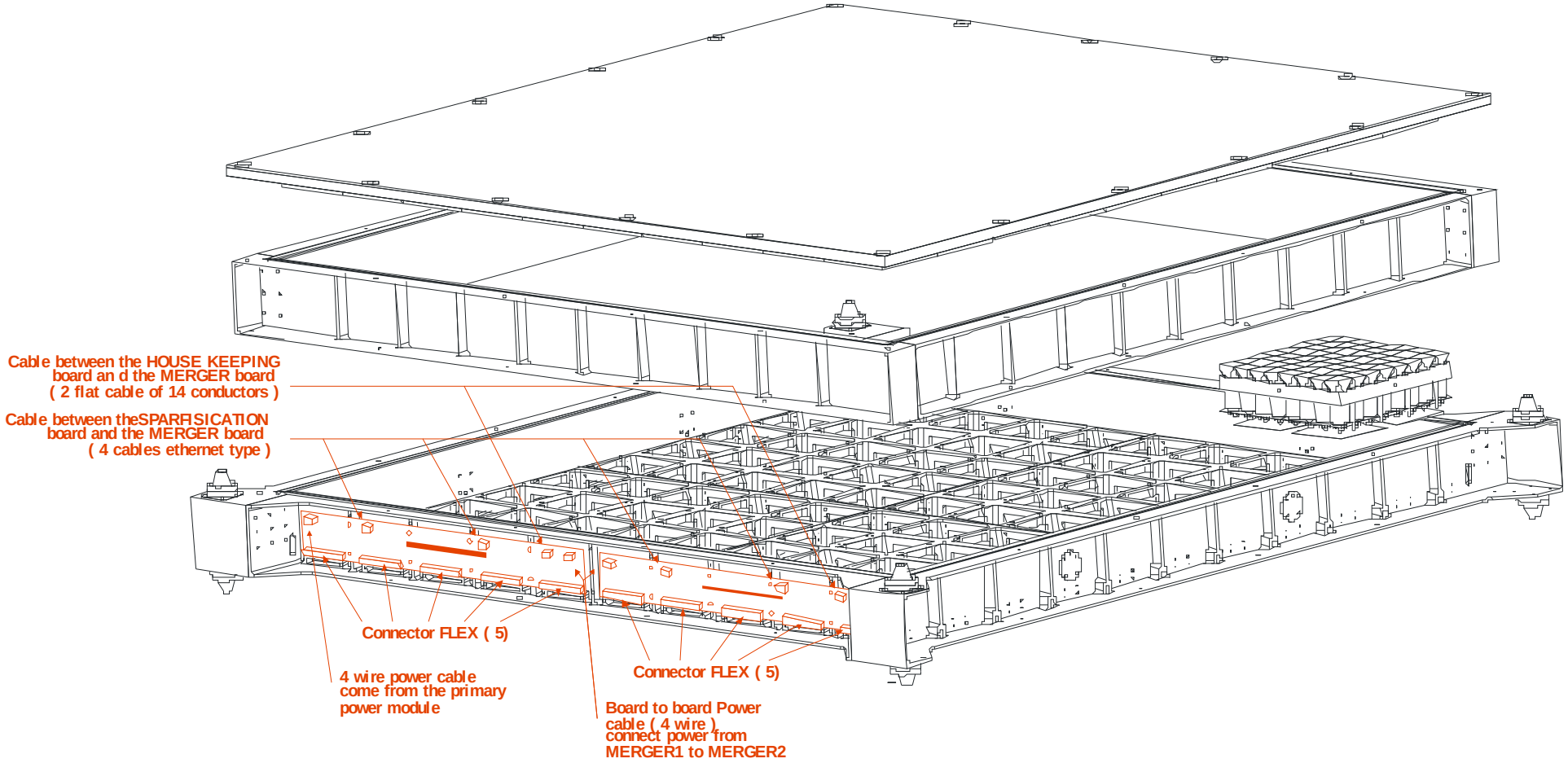
TRIGGER with LED ON



TRIGGER with LED OFF

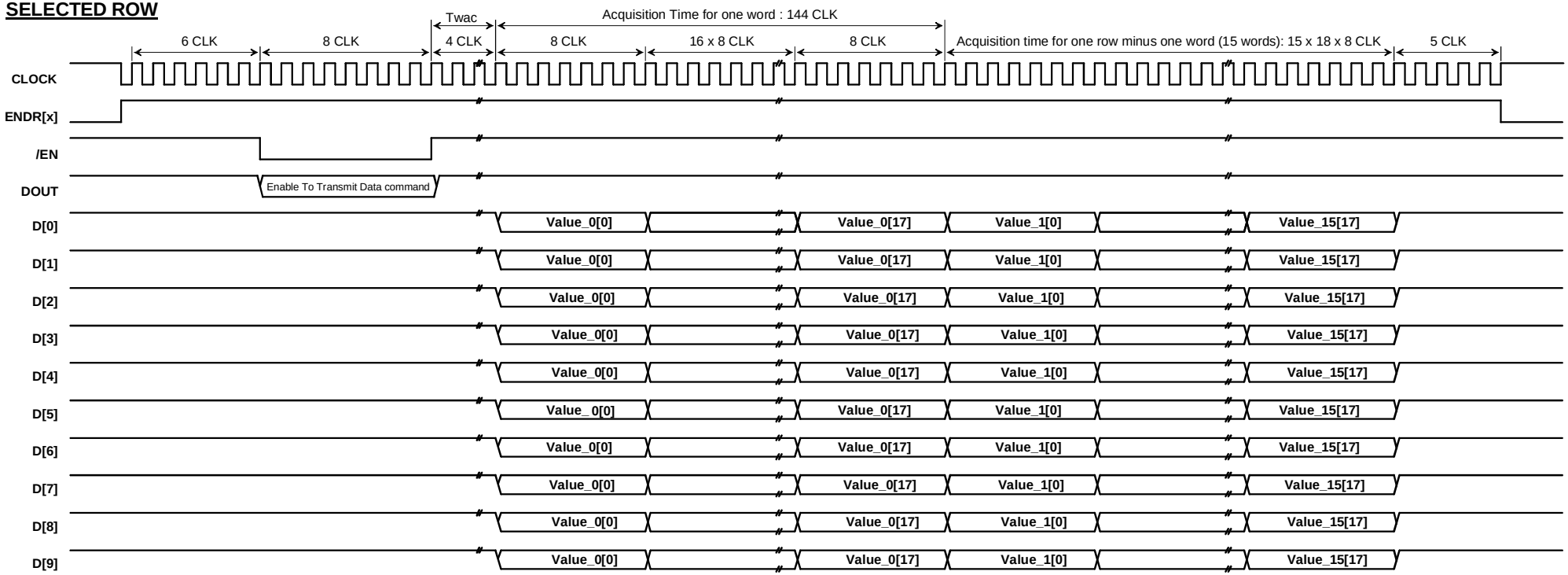


Annex 5 : boards implantation on the detector

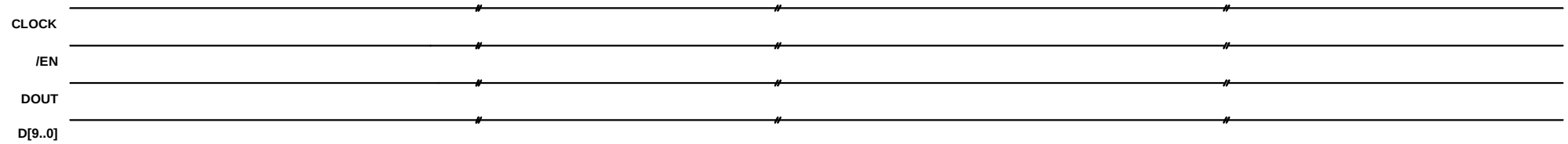


Annex 6 : Row Data Acquisition Sequence

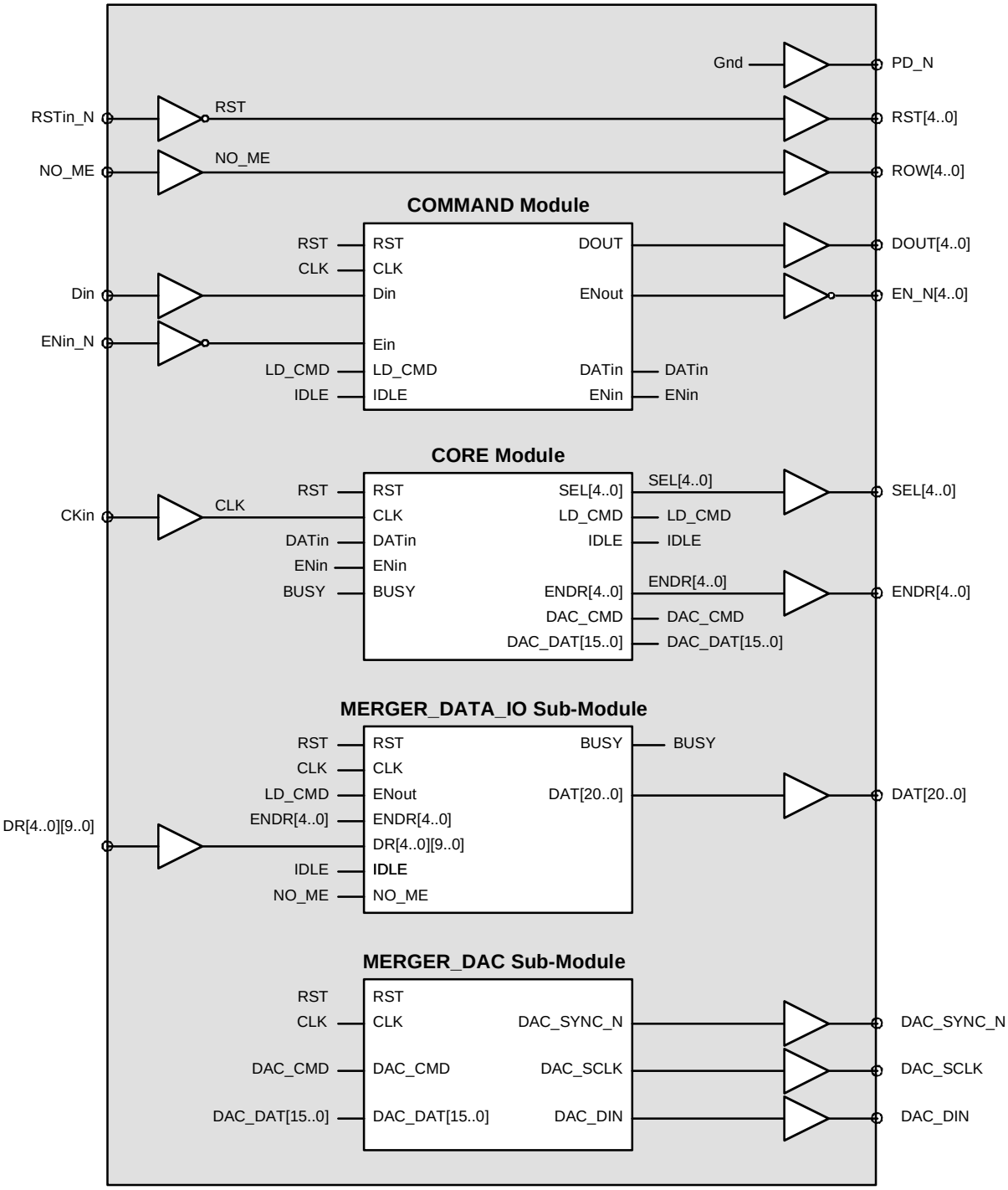
SELECTED ROW



DISABLED ROW



Annex 7 : “MERGER” FPGA synoptic with 21 bits SERDES



"MERGER" FPGA
August 10 2006
Data Acquisition
Laboaratoire de Physique Subatomique et de Cosmologie
Grenoble, FRANCE

Annex 8 : MAIN and SLAVE MERGER_CORE sub-module architecture

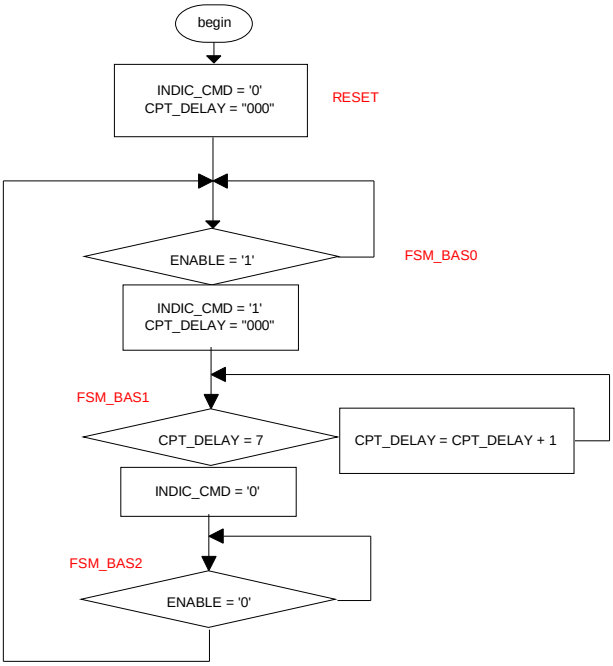


Figure 5 : MAIN MERGER_CORE sub module

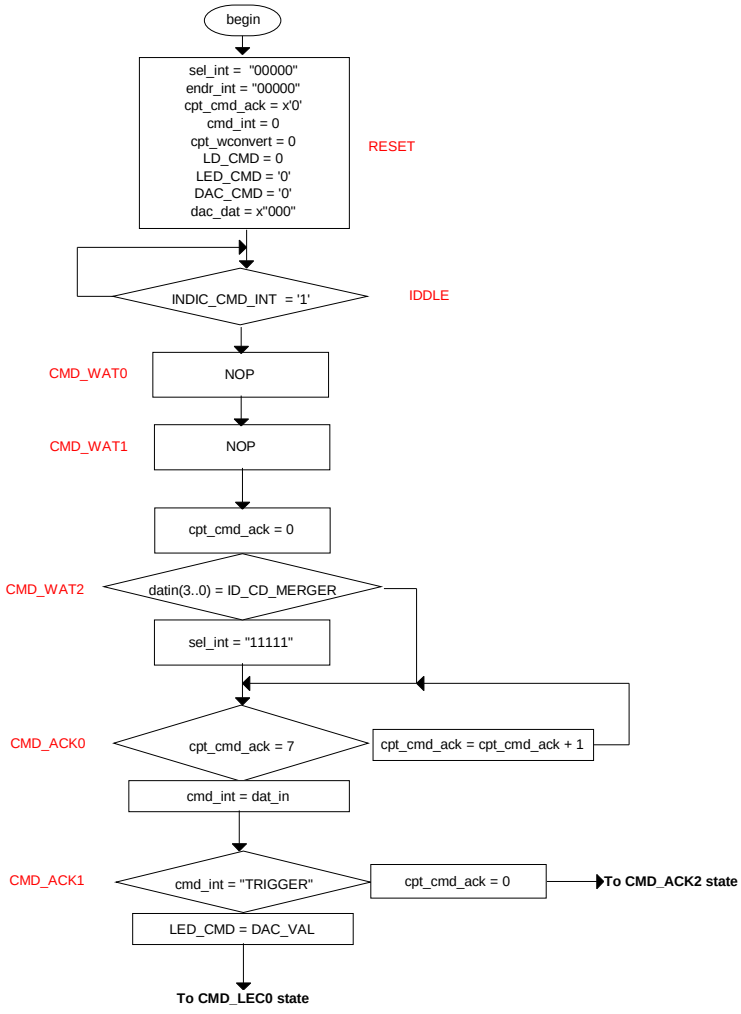


Figure 6 : SLAVE MERGER_CORE sub module (1/3)

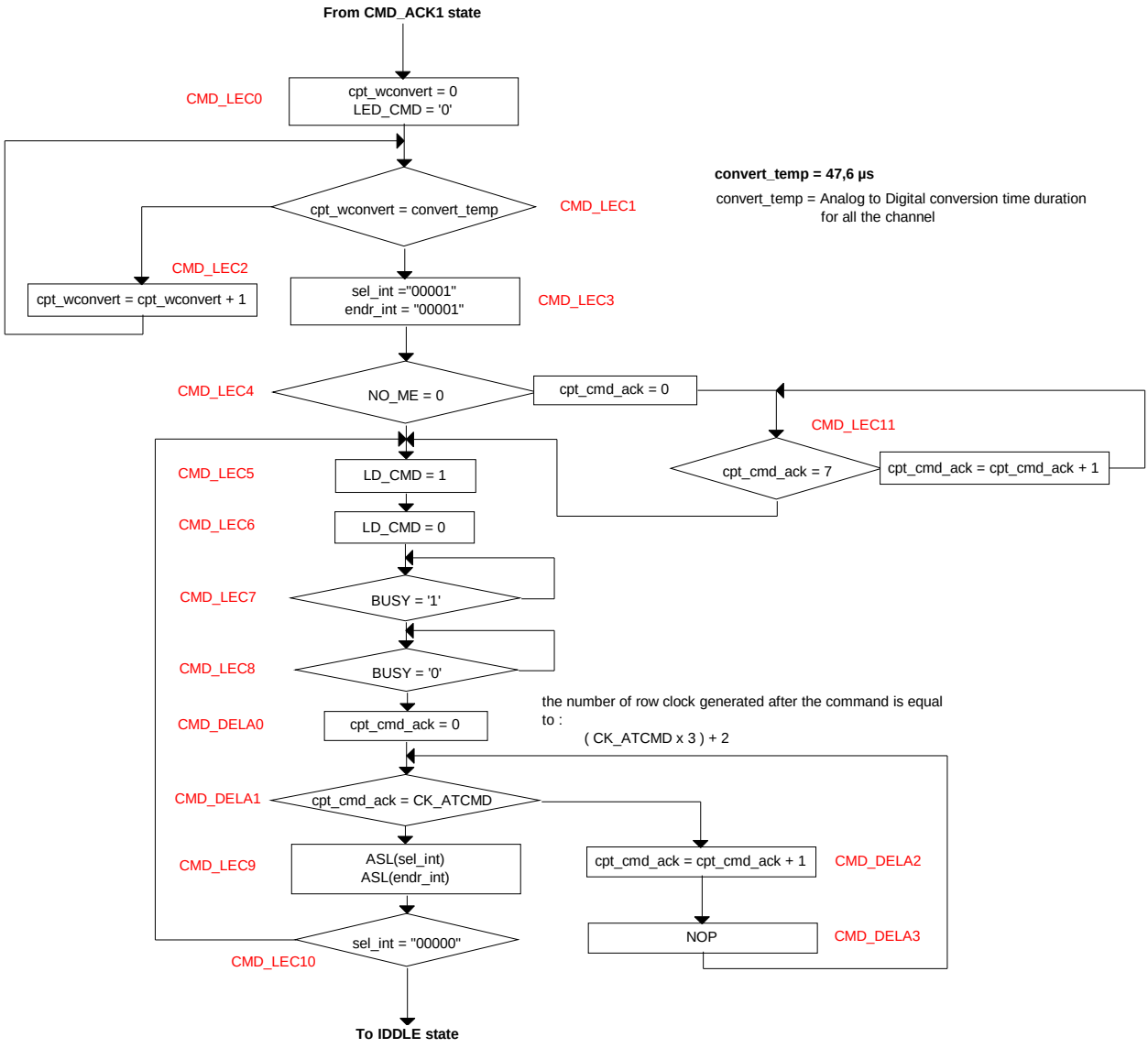


Figure 7 : SLAVE MERGER_CORE sub module (2/3)

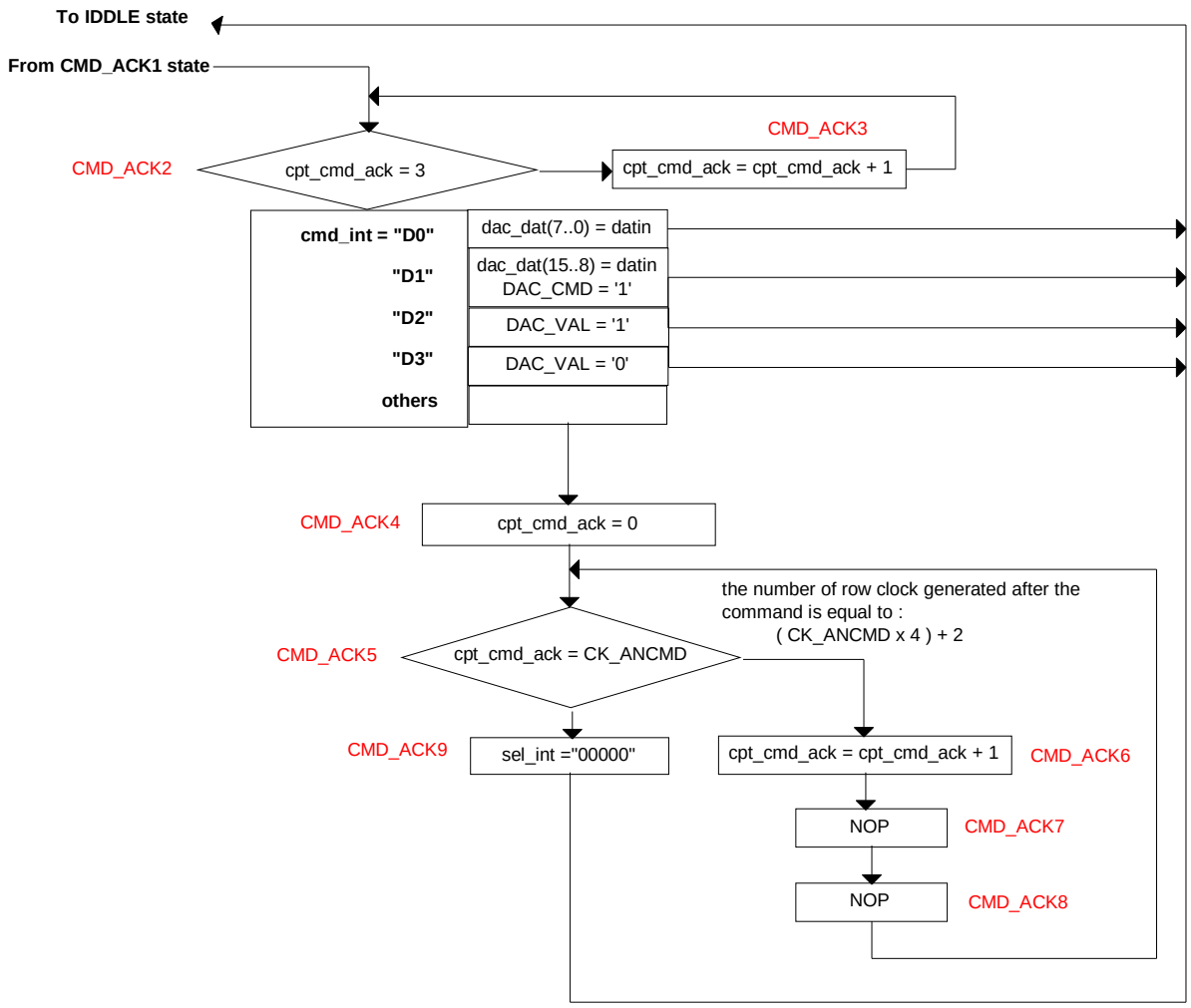


Figure 8 : SLAVE MERGER_CORE sub module (3/3)

Annex 9 : MERGER_DATA_IO sub-module architecture

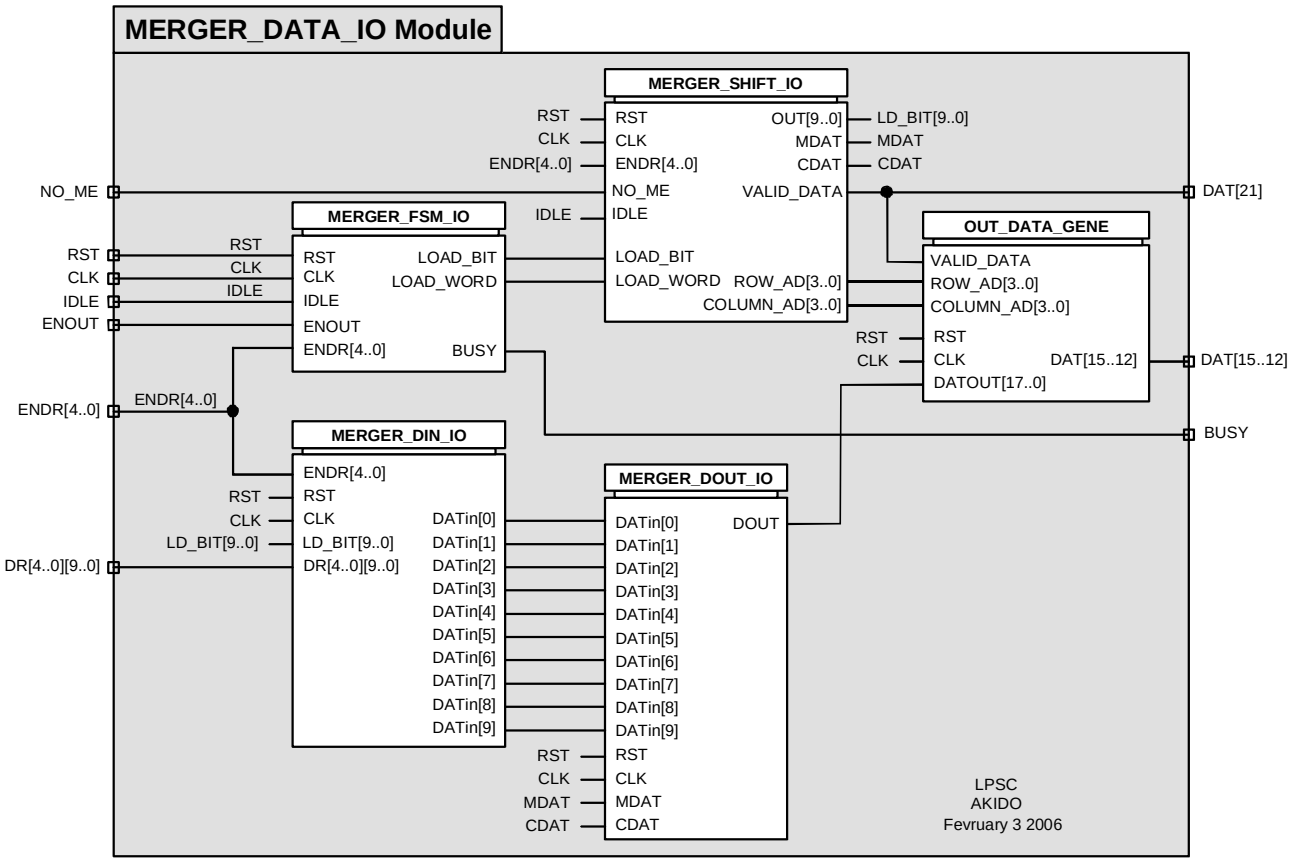


Figure 9 : IO_data module architecture

The following scheme show a part of the MERGER_DIN_IO module. This one contains 10 element of the following picture.

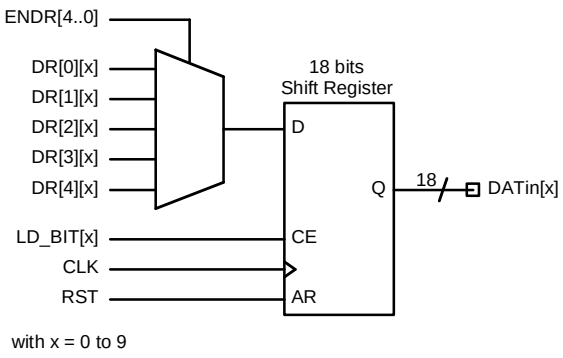


Figure 10 : MERGER_DIN_IO elementary sub module

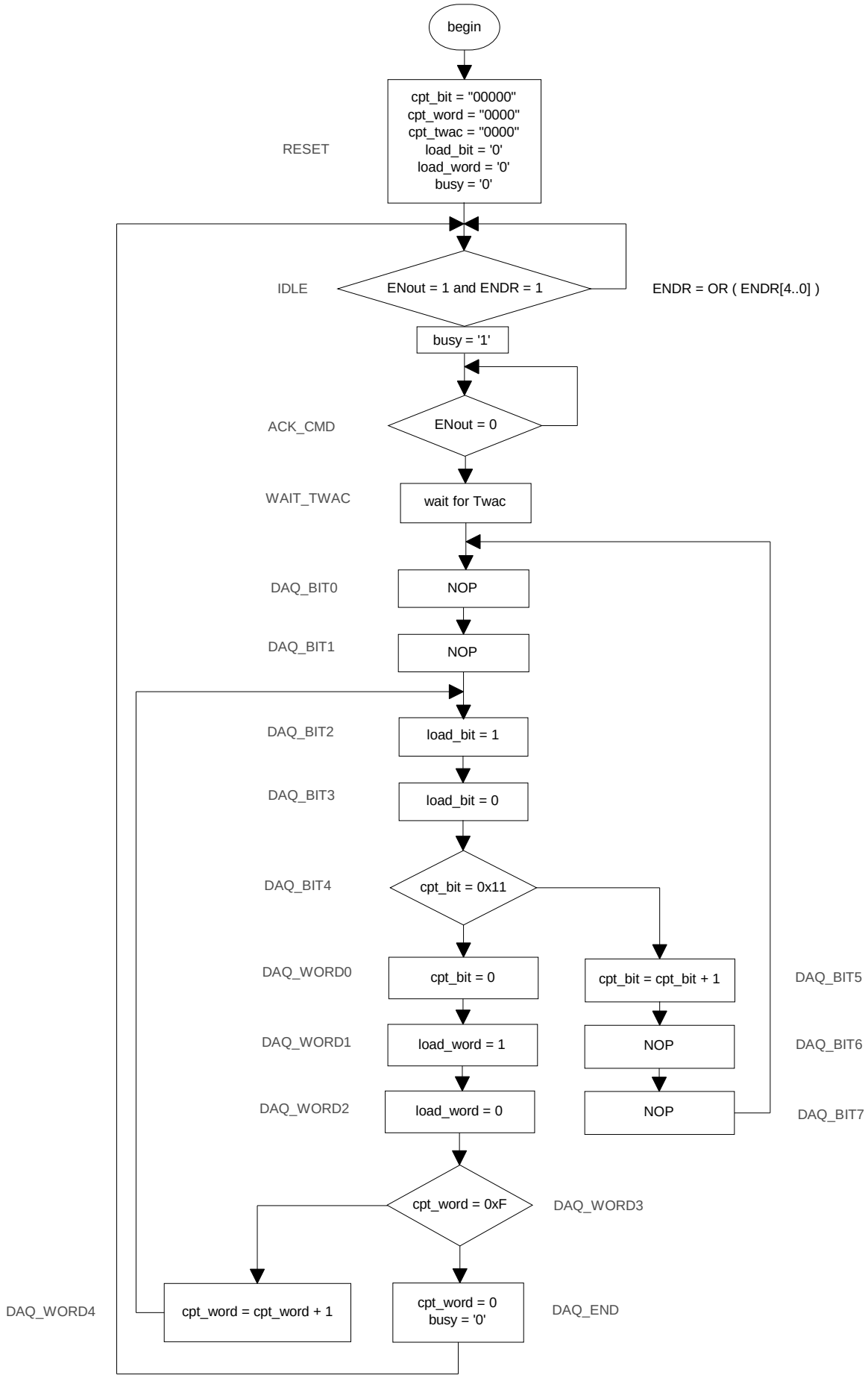


Figure 11 : Merger_FSM_IO module architecture

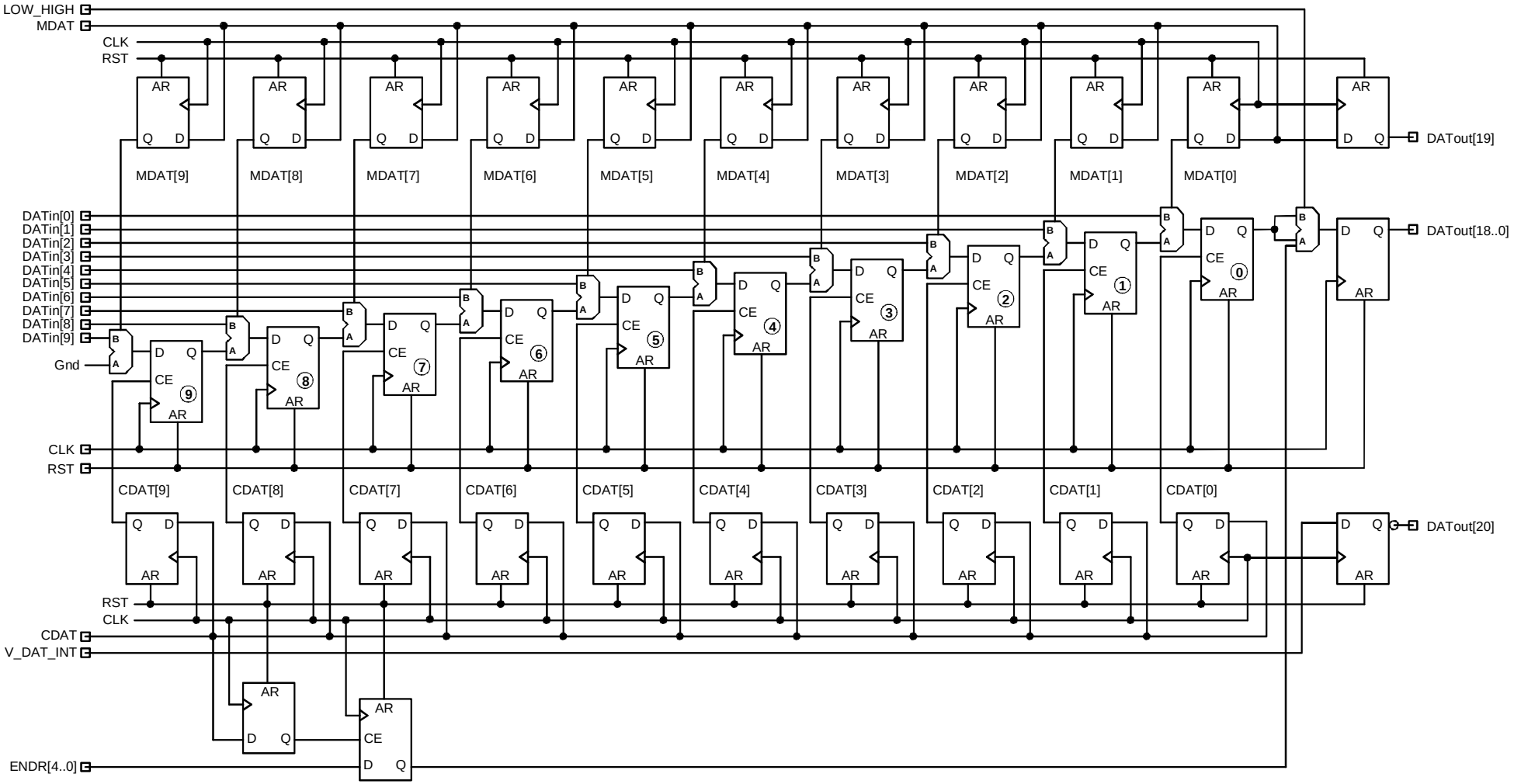


Figure 12 : MERGER_DOUT_IO Module architecture

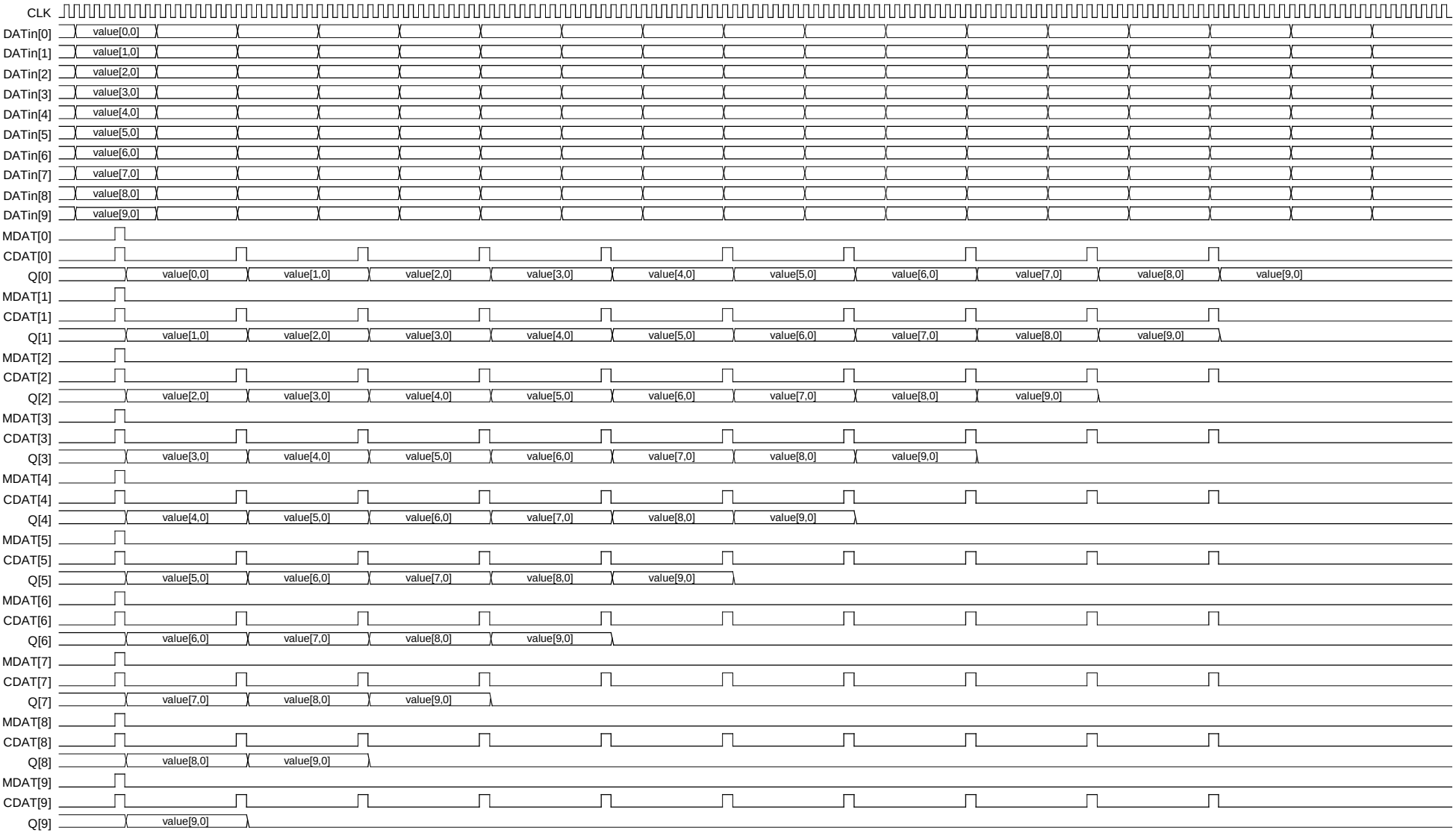


Figure 13 : MERGER_DOUT_IO Module timing diagram

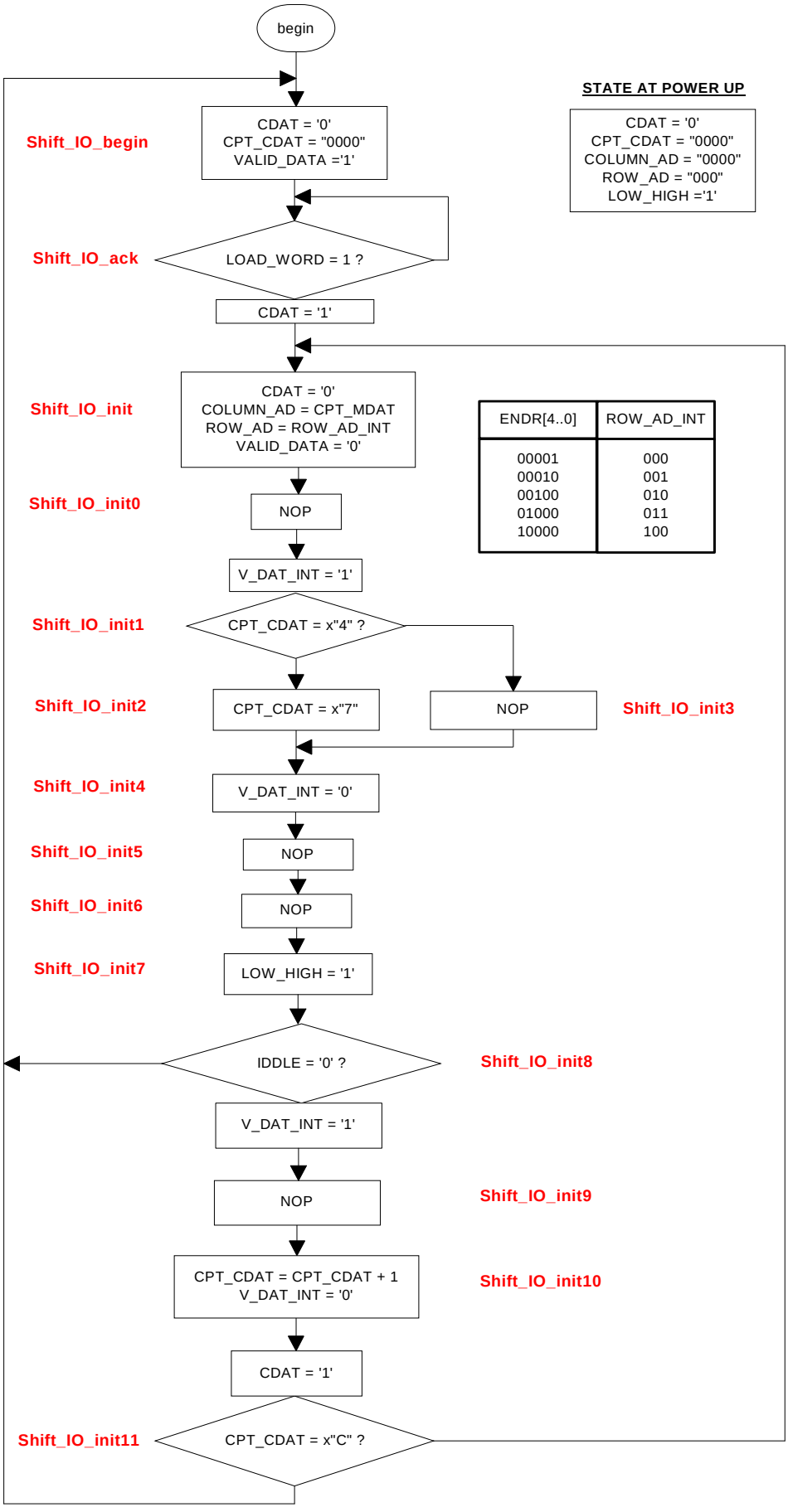


Figure 14 : Merger_Shift_IO FSM module architecture

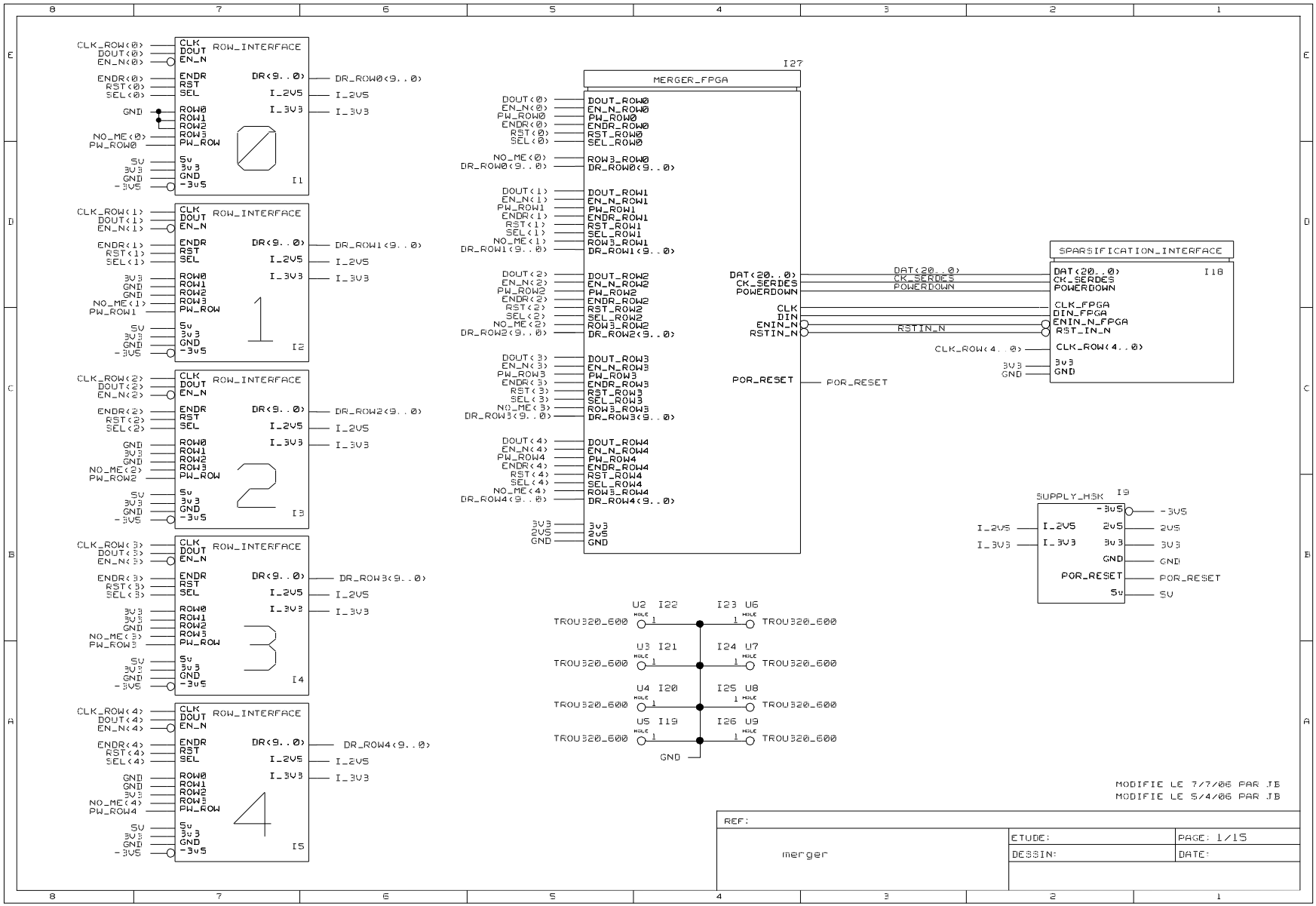
Annex 10 : SPARSIFICATION up LINK connector pin out

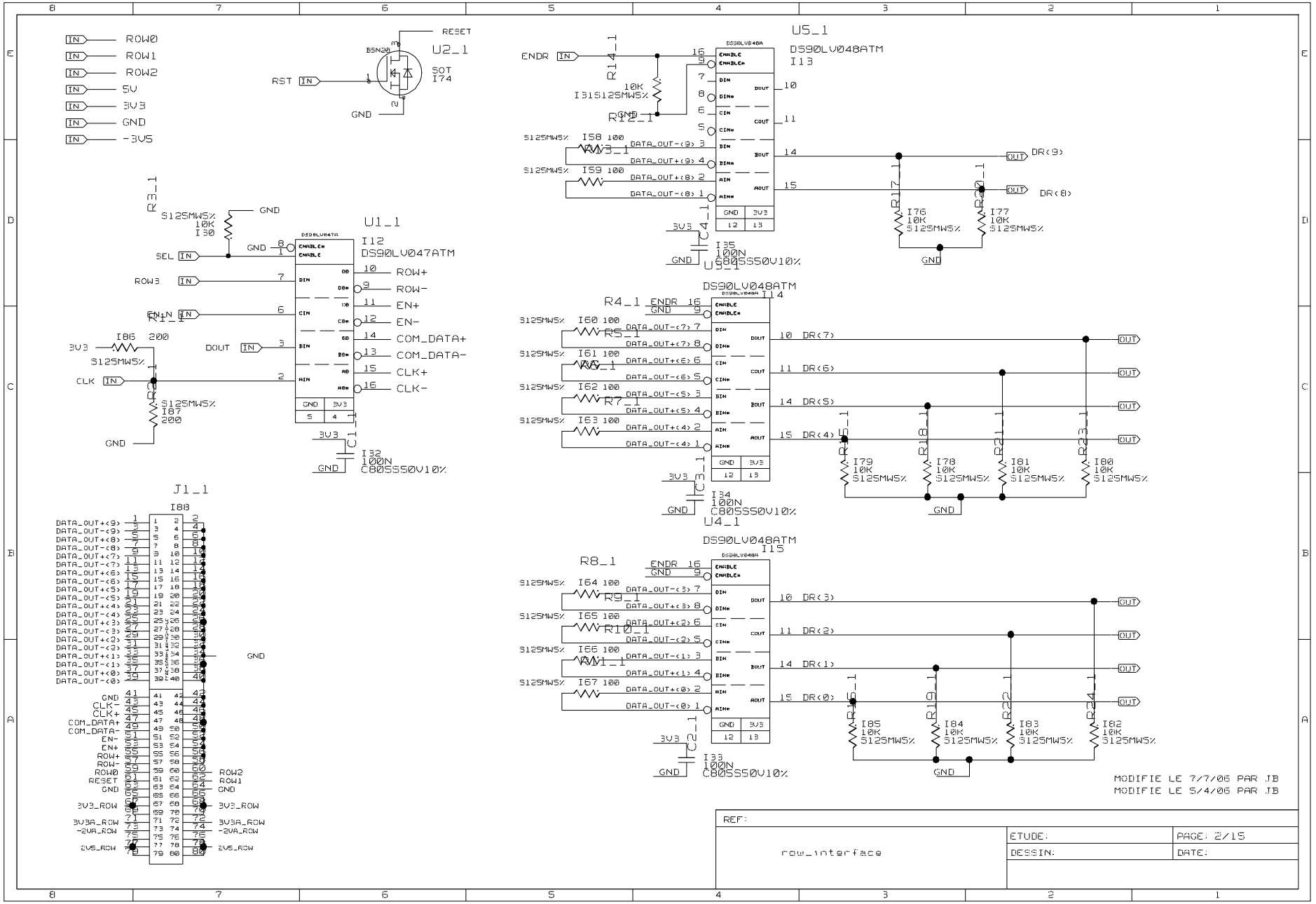
Pin Number	Name signal
1	LVDS_DATA_RX+
2	LVDS_DATA_RX-
3	LVDS_EN_RX+
4	LVDS_NRST_RX-
5	LVDS_NRST_RX+
6	LVDS_EN_RX-
7	LVDS_CLK_RX+
8	LVDS_CLK_RX+

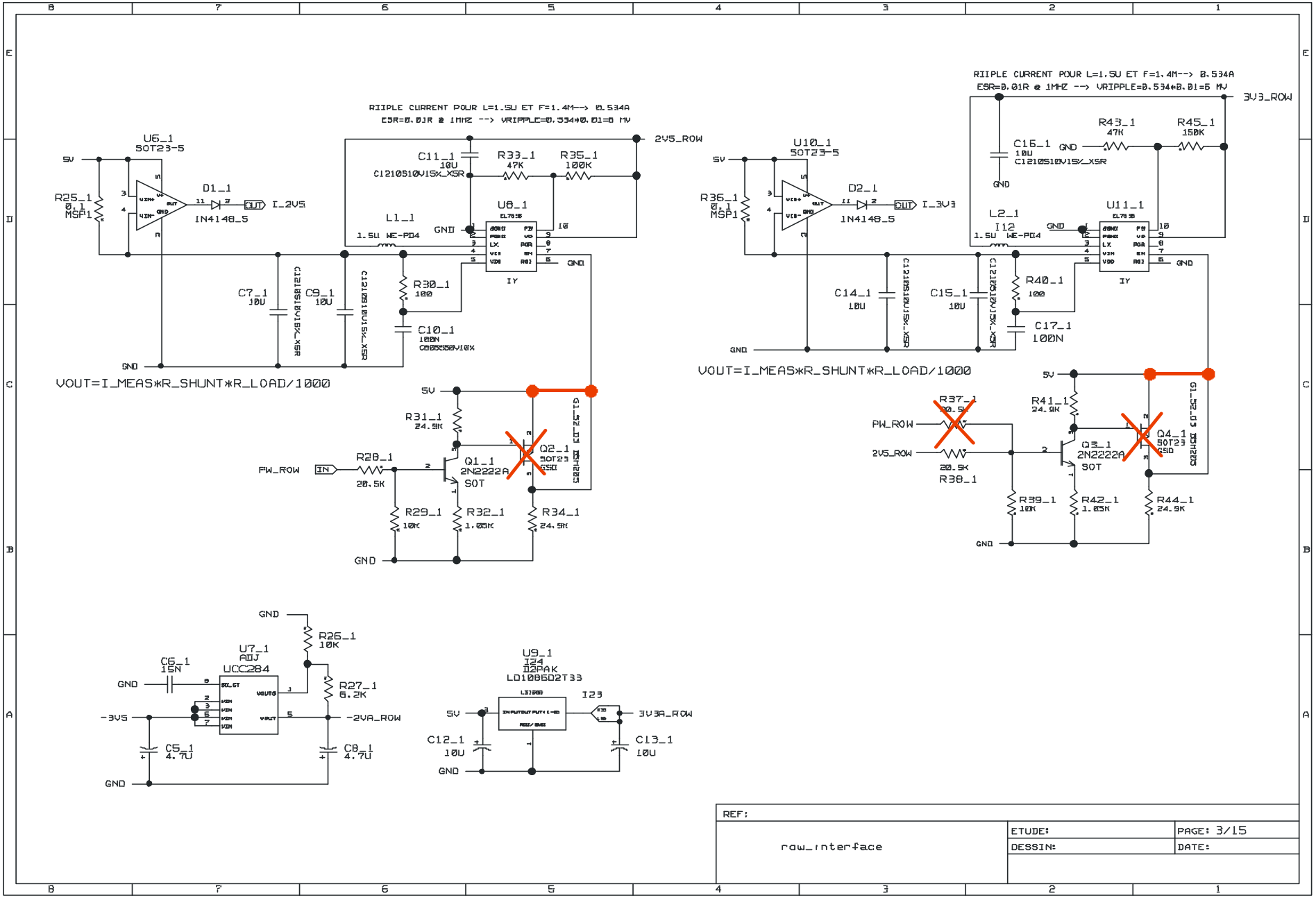
Annex 11 : SPARSIFICATION down LINK connector pin out

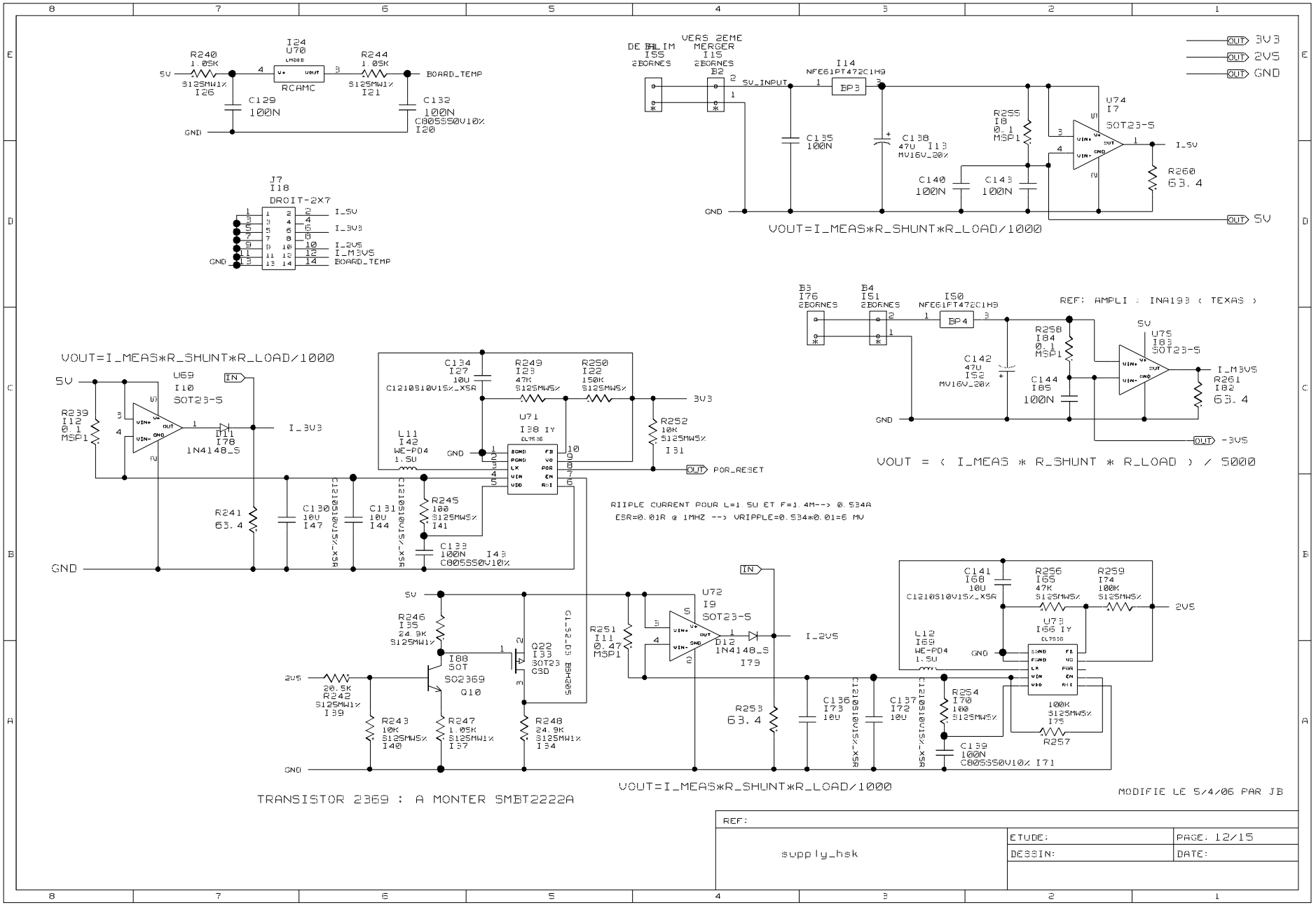
Pin Number	Name signal
1	LVDS_TX0-
2	LVDS_ TX0+
3	LVDS_ TX1-
4	LVDS_ TX2+
5	LVDS_ TX2-
6	LVDS_ TX1+
7	LVDS_TXCLK-
8	LVDS_ TXCLK+

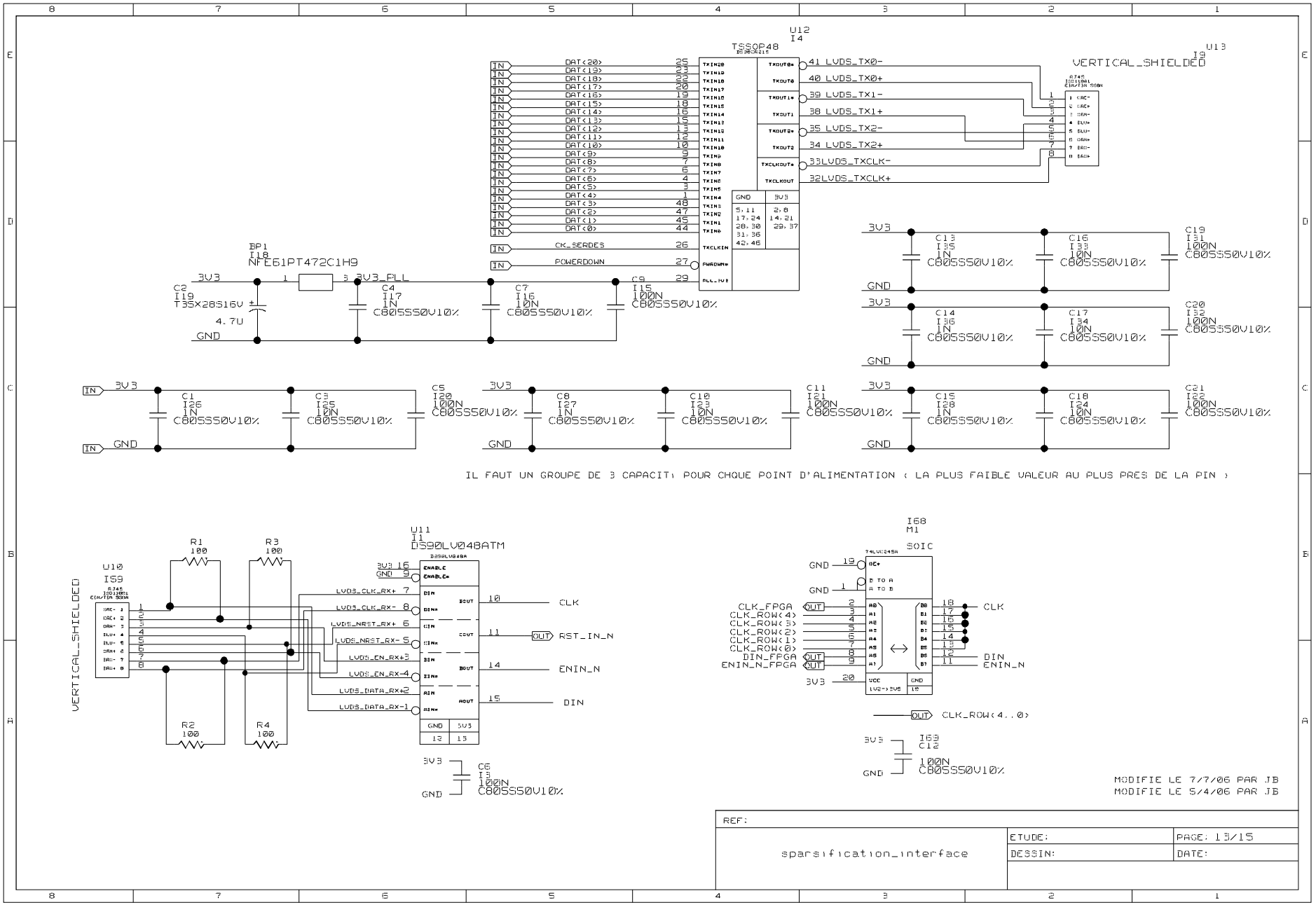
Annex 12 : Board scheme

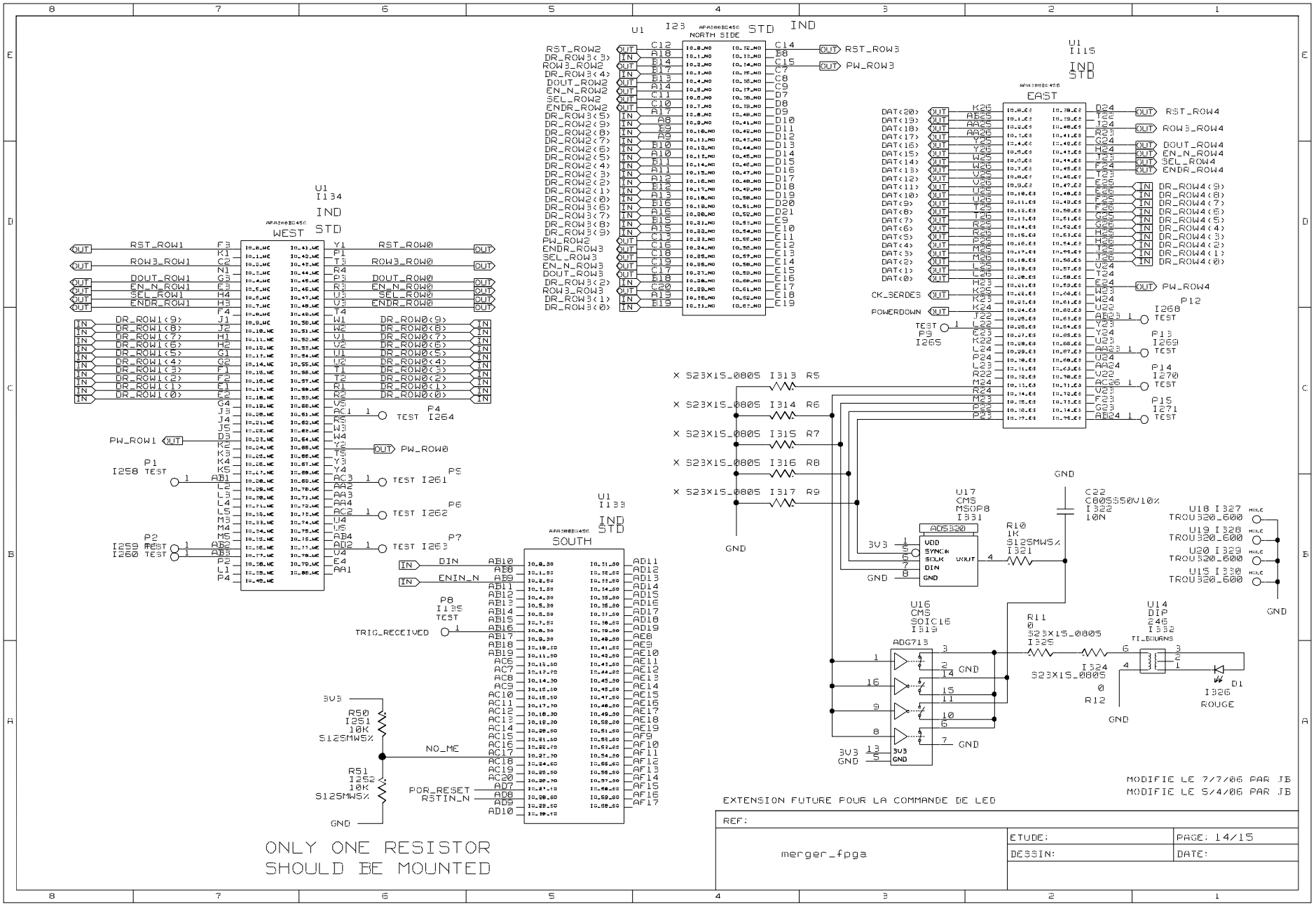


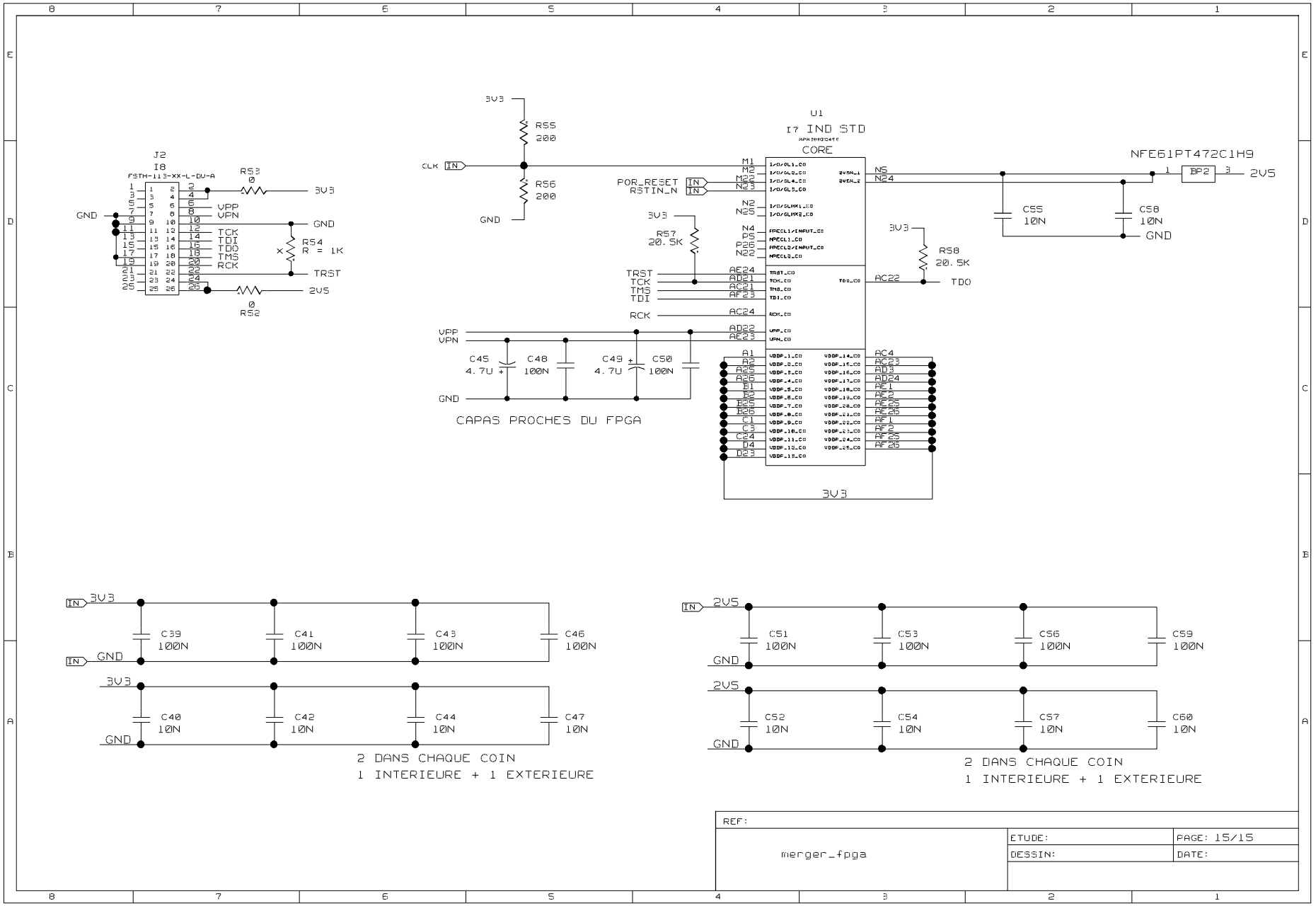

ROW0
ROW1
ROW2
5V
3V3
GND
-3V5
RESET
U2_1
SOT
I74
RST
GND
U1_1
I12
DS90LV047ATM
SEL
ROW3
EN+N
DOUT
CLK
ROW+
ROW-
EN+
EN-
COM_DATA+
COM_DATA-
CLK+
CLK-
U5_1
DS90LV048ATM
I13
ENDR
DR<9>
DR<8>
U3_1
U4_1
I14
DR<7>
DR<6>
DR<5>
DR<4>
U2_1
I15
DR<3>
DR<2>
DR<1>
DR<0>
J1_1
I88
ROW2
ROW1
3V3_ROW
3V3A_ROW
-2VA_ROW
2V5_ROW
MODIFIE LE 7/7/06 PAR JB
MODIFIE LE 5/4/06 PAR JB
REF:
row_interface
ETUDE:
PAGE: 2/15
DESSIN:
DATE:


RIIPLE CURRENT POUR L=1.5U ET F=1.4M--> 0.534A
ESR=0.01R @ 1MHZ --> VRIPPLE=0.534*0.01=5 MV
U6_1
SOT23-5
5V
R25_1
0.1
MSP1
D1_1
IN4148_5
I_2V5
C11_1
10U
C1210S10V15%_X5R
R33_1
47K
R35_1
100K
2V5_ROW
L1_1
1.5U WE-PD4
GND
U8_1
EL7535
C7_1
10U
C9_1
10U
R30_1
100
C10_1
100N
C0805S50V10%
VOUT=I_MEAS*R_SHUNT*R_LOAD/1000
R31_1
24.9K
R28_1
20.5K
FW_ROW
IN
Q1_1
2N2222A
SOT
Q2_1
SOT23
BSH205
R29_1
10K
R32_1
1.05K
R34_1
24.9K
U10_1
SOT23-5
R36_1
0.1
MSP1
D2_1
IN4148_5
I_3V3
L2_1
I12
1.5U WE-PD4
U11_1
EL7535
C14_1
10U
C15_1
10U
R40_1
100
C17_1
100N
C16_1
10U
C1210S10V15%_X5R
R43_1
47K
R45_1
150K
3V3_ROW
R37_1
20.5K
PW_ROW
2V5_ROW
20.5K
R38_1
R41_1
24.9K
Q3_1
2N2222A
SOT
Q4_1
SOT23
BSH205
R39_1
10K
R42_1
1.05K
R44_1
24.9K
U7_1
ADJ
UCC284
C6_1
15N
R26_1
10K
R27_1
6.2K
-3V5
-2V5A_ROW
C5_1
4.7U
C8_1
4.7U
U9_1
I24
D2PAK
LD1086D2T33
I23
3V3A_ROW
C12_1
10U
C13_1
10U
REF:
raw_interface
ETUDE:
DESSIN:
PAGE: 3/15
DATE:

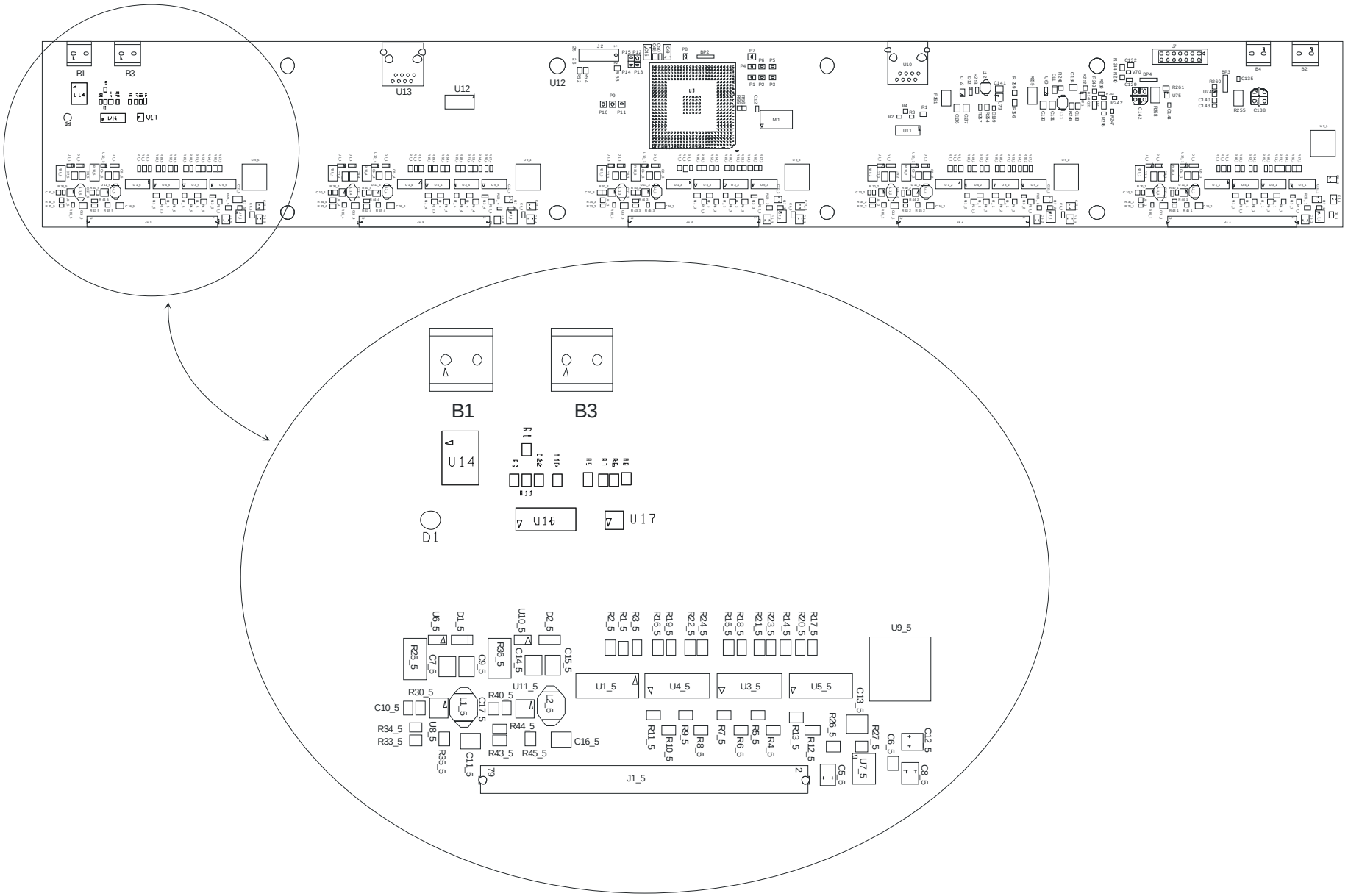

VOUT=I_MEAS*R_SHUNT*R_LOAD/1000
VOUT = (I_MEAS * R_SHUNT * R_LOAD) / 5000
RIIPLE CURRENT POUR L=1.5U ET F=1.4M--> 0.534A
ESR=0.01R @ 1MHZ --> VRIPPLE=0.534*0.01=6 MV
TRANSISTOR 2369 : A MONTER SMBT2222A
MODIFIE LE 5/4/06 PAR JB
REF: AMPLI : INA193 (TEXAS)
supply_hsk
ETUDE:
PAGE: 12/15
DESSIN:
DATE:

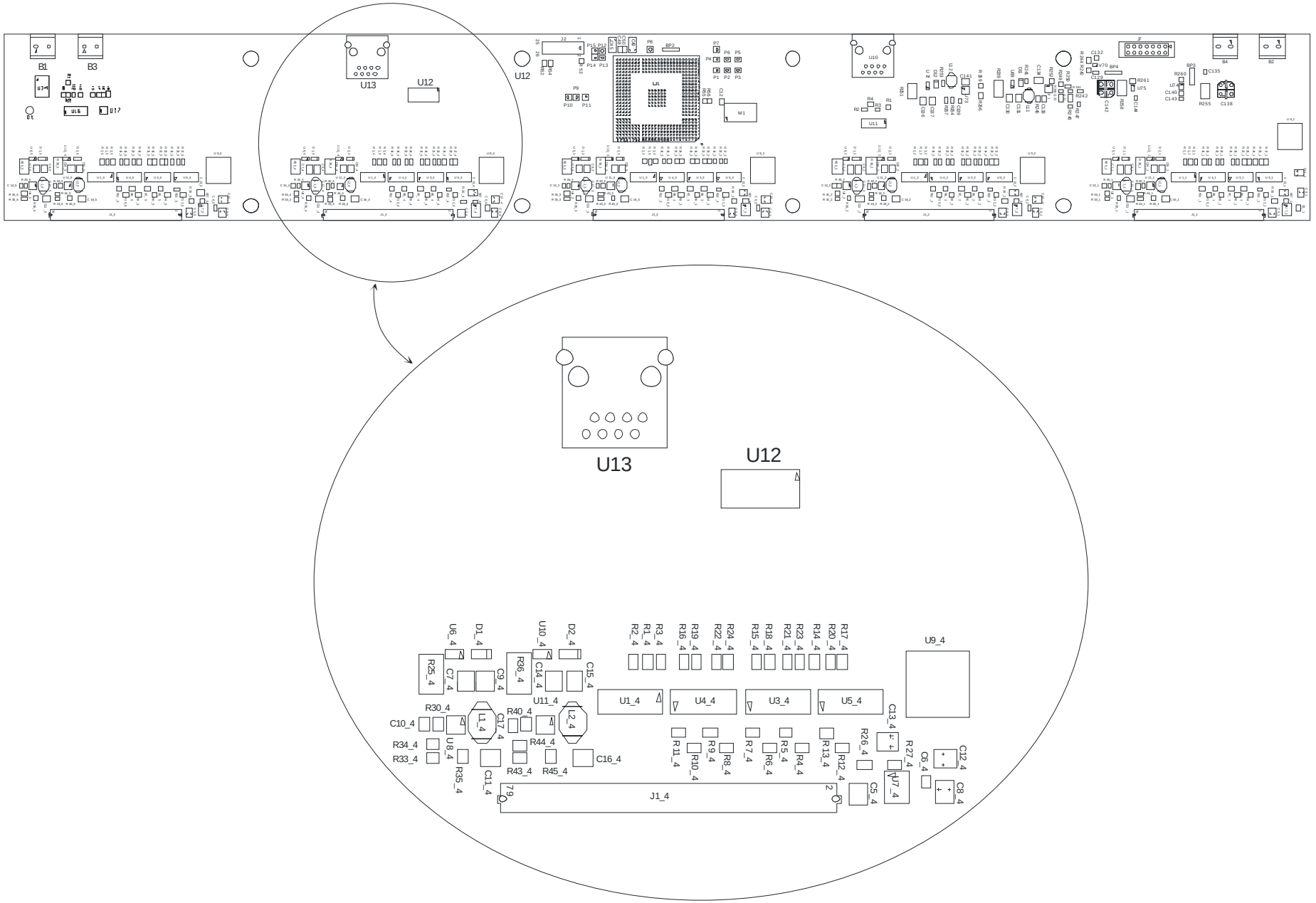

U12
I4
TSSOP48
U13
I9
VERTICAL_SHIELDED
DAT<20>
DAT<19>
DAT<18>
DAT<17>
DAT<16>
DAT<15>
DAT<14>
DAT<13>
DAT<12>
DAT<11>
DAT<10>
DAT<9>
DAT<8>
DAT<7>
DAT<6>
DAT<5>
DAT<4>
DAT<3>
DAT<2>
DAT<1>
DAT<0>
CK_SERDES
POWERDOWN
41 LVDS_TX0-
40 LVDS_TX0+
39 LVDS_TX1-
38 LVDS_TX1+
35 LVDS_TX2-
34 LVDS_TX2+
33LVDS_TXCLK-
32LVDS_TXCLK+
GND
3V3
BP1
I18
NFE61PT472C1H9
3V3_PLL
C2
I19
T35X285I6V ±
4.7U
C4
I17
1N
C805S50V10%
C7
I16
10N
C805S50V10%
C9
I15
100N
C805S50V10%
C13
I35
1N
C16
I33
10N
C19
I31
100N
C14
I36
1N
C17
I34
10N
C20
I32
100N
C1
I26
1N
C3
I25
10N
C5
I20
100N
C8
I27
1N
C10
I23
10N
C11
I21
100N
C15
I28
1N
C18
I24
10N
C21
I22
100N
IL FAUT UN GROUPE DE 3 CAPACITÉ POUR CHQUE POINT D'ALIMENTATION (LA PLUS FAIBLE VALEUR AU PLUS PRES DE LA PIN)
U10
I59
VERTICAL_SHIELDED
R1
100
R3
100
R2
100
R4
100
U11
I1
DS90LV048ATM
LVDS_CLK_RX+
LVDS_CLK_RX-
LVDS_NRST_RX+
LVDS_NRST_RX-
LVDS_EN_RX+3
LVDS_EN_RX-4
LVDS_DATA_RX+2
LVDS_DATA_RX-1
CLK
RST_IN_N
ENIN_N
DIN
C6
I3
100N
C805S50V10%
I68
M1
SOIC
CLK_FPGA
CLK_ROW<4>
CLK_ROW<3>
CLK_ROW<2>
CLK_ROW<1>
CLK_ROW<0>
DIN_FPGA
ENIN_N_FPGA
CLK_ROW<4..0>
I69
C12
100N
C805S50V10%
MODIFIE LE 7/7/06 PAR JB
MODIFIE LE 5/4/06 PAR JB
REF:
sparsification_interface
ETUDE:
PAGE: 13/15
DESSIN:
DATE:

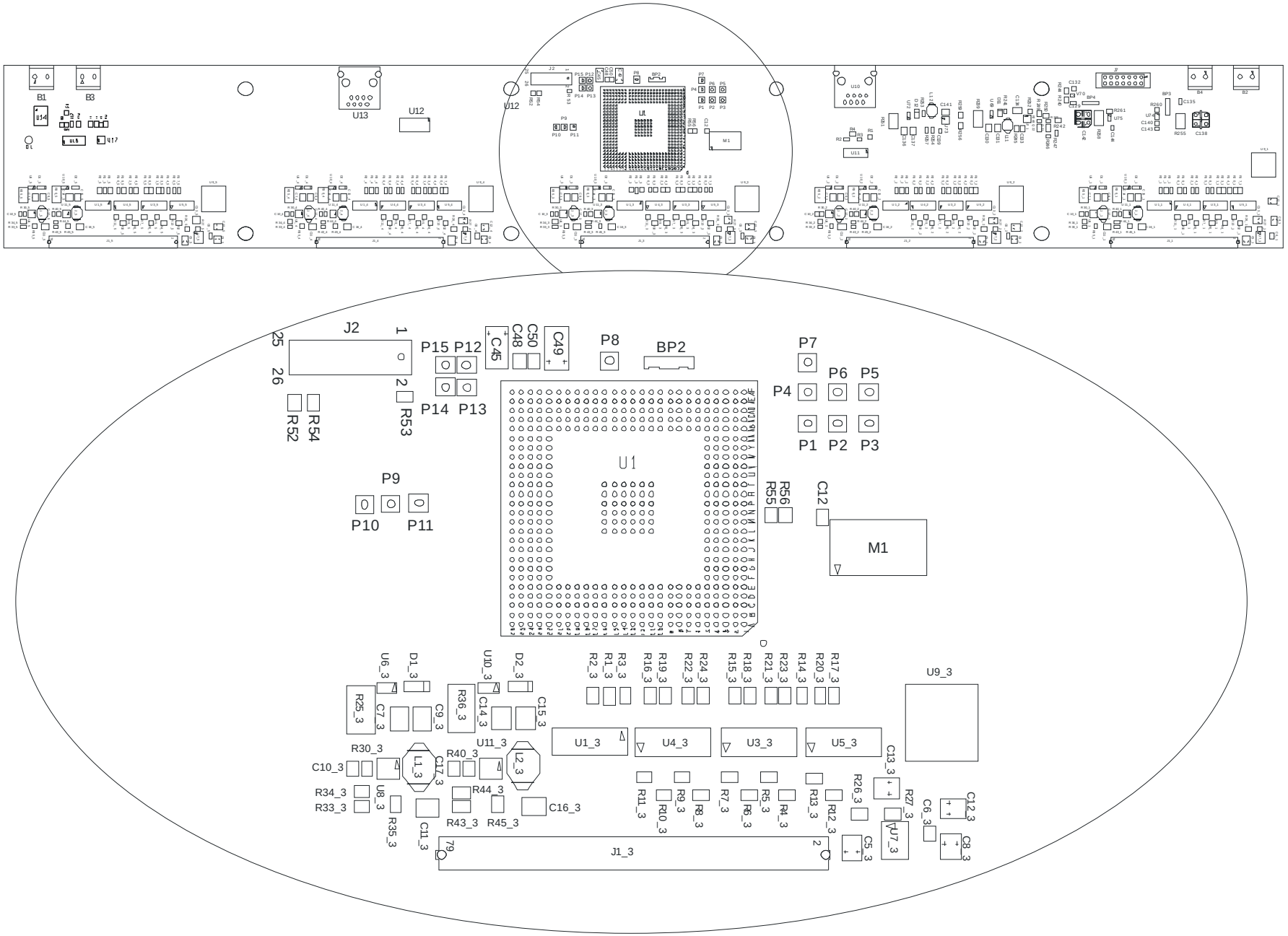

U1 I23 NORTH SIDE STD IND
RST_ROW2
DR_ROW3<3>
ROW3_ROW2
DR_ROW3<4>
DOUT_ROW2
EN_N_ROW2
SEL_ROW2
ENDR_ROW2
DR_ROW3<5>
DR_ROW2<9>
DR_ROW2<8>
DR_ROW2<7>
DR_ROW2<6>
DR_ROW2<5>
DR_ROW2<4>
DR_ROW2<3>
DR_ROW2<2>
DR_ROW2<1>
DR_ROW2<0>
DR_ROW3<6>
DR_ROW3<7>
DR_ROW3<8>
DR_ROW3<9>
PW_ROW2
ENDR_ROW3
SEL_ROW3
EN_N_ROW3
DOUT_ROW3
DR_ROW3<2>
ROW3_ROW3
DR_ROW3<1>
DR_ROW3<0>
RST_ROW3
PW_ROW3
U1 I115 IND STD EAST
DAT<20>
DAT<19>
DAT<18>
DAT<17>
DAT<16>
DAT<15>
DAT<14>
DAT<13>
DAT<12>
DAT<11>
DAT<10>
DAT<9>
DAT<8>
DAT<7>
DAT<6>
DAT<5>
DAT<4>
DAT<3>
DAT<2>
DAT<1>
DAT<0>
CK_SERDES
POWERDOWN
TEST P9 I265
RST_ROW4
ROW3_ROW4
DOUT_ROW4
EN_N_ROW4
SEL_ROW4
ENDR_ROW4
DR_ROW4<9>
DR_ROW4<8>
DR_ROW4<7>
DR_ROW4<6>
DR_ROW4<5>
DR_ROW4<4>
DR_ROW4<3>
DR_ROW4<2>
DR_ROW4<1>
DR_ROW4<0>
PW_ROW4
P12 I268 TEST
P13 I269 TEST
P14 I270 TEST
P15 I271 TEST
U1 I134 IND STD WEST
RST_ROW1
ROW3_ROW1
DOUT_ROW1
EN_N_ROW1
SEL_ROW1
ENDR_ROW1
DR_ROW1<9>
DR_ROW1<8>
DR_ROW1<7>
DR_ROW1<6>
DR_ROW1<5>
DR_ROW1<4>
DR_ROW1<3>
DR_ROW1<2>
DR_ROW1<1>
DR_ROW1<0>
RST_ROW0
ROW3_ROW0
DOUT_ROW0
EN_N_ROW0
SEL_ROW0
ENDR_ROW0
DR_ROW0<9>
DR_ROW0<8>
DR_ROW0<7>
DR_ROW0<6>
DR_ROW0<5>
DR_ROW0<4>
DR_ROW0<3>
DR_ROW0<2>
DR_ROW0<1>
DR_ROW0<0>
P4 TEST I264
PW_ROW1
PW_ROW0
P1 I258 TEST
P2 I259 TEST I260 TEST
P5 TEST I261
P6 TEST I262
P7 TEST I263
P8 I135 TEST
DIN
ENIN_N
TRIG_RECEIVED
U1 I133 IND STD SOUTH
3V3
R50 I251 10K S125MW5%
R51 I252 10K S125MW5%
NO_ME
POR_RESET
RSTIN_N
GND
ONLY ONE RESISTOR SHOULD BE MOUNTED
X S23X15_0805 I313 R5
X S23X15_0805 I314 R6
X S23X15_0805 I315 R7
X S23X15_0805 I316 R8
X S23X15_0805 I317 R9
U17 CMS MSOP8 I331
AD5320
VDD
SYNC
SCLK
DIN
VOUT
C22 C805550V10% I322 10N
R10 1K S125MW5% I321
U18 I327 TROU320_600
U19 I328 TROU320_600
U20 I329 TROU320_600
U15 I330 TROU320_600
U16 CMS SOIC16 I319
AD713
R11 0 S23X15_0805 I325
R12 0 S23X15_0805 I324
U14 DIP 246 I332 TI_BOURNS
D1 I326 ROUGE
MODIFIE LE 7/7/06 PAR JB
MODIFIE LE 5/4/06 PAR JB
EXTENSION FUTURE POUR LA COMMANDE DE LED
REF:
merger_fpga
ETUDE:
PAGE: 14/15
DESSIN:
DATE:

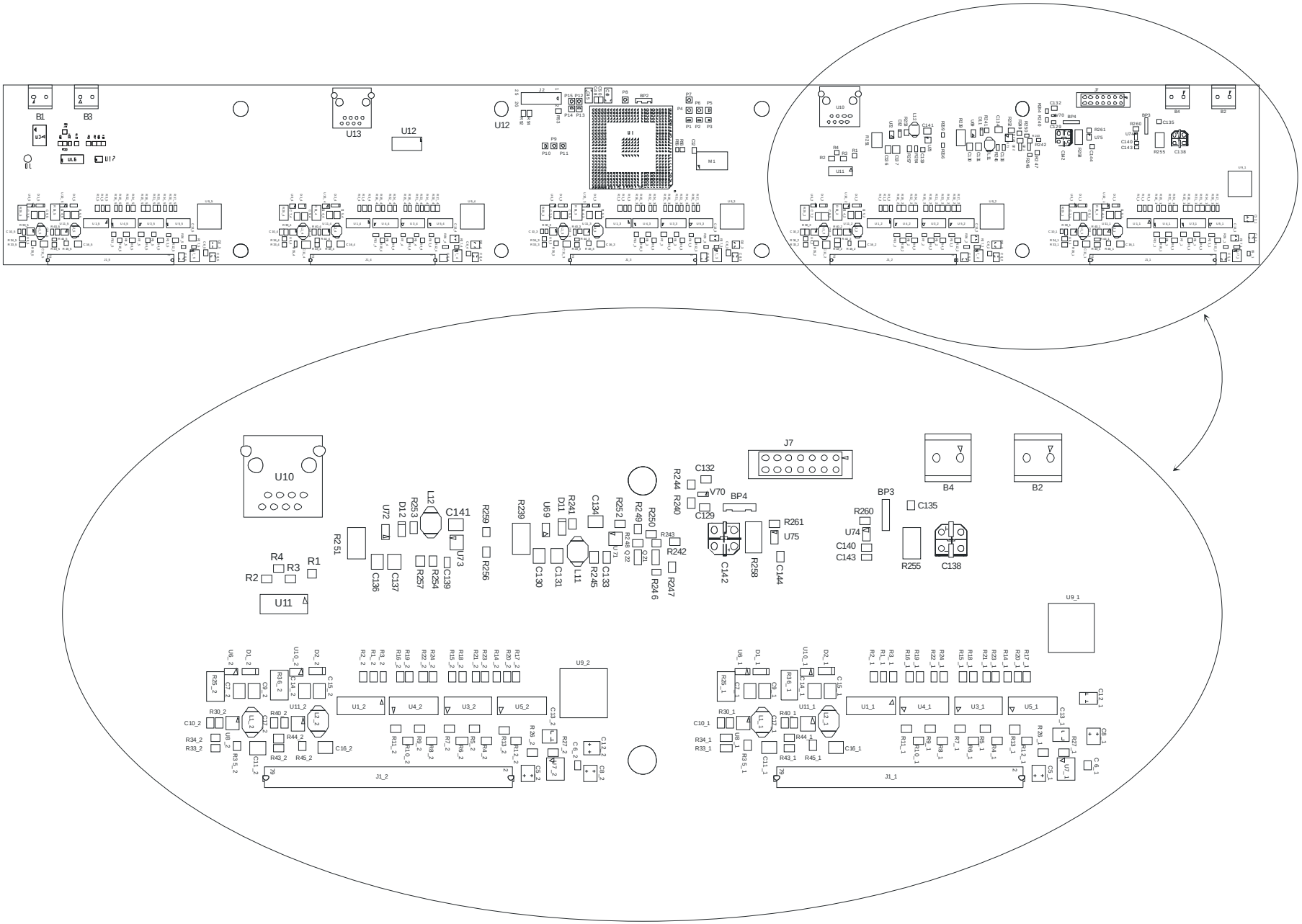

J2
I8
FSTH-113-XX-L-DV-A
RS3
0
3V3
VPP
VPN
GND
TCK
TDI
TDO
TMS
RCK
TRST
2V5
RS4
R = 1K
RS2
U1
I7 IND STD
CORE
NFE61PT472C1H9
BP2
CLK
RS5
200
RS6
200
POR_RESET
RSTIN_N
RS7
20.5K
RS8
20.5K
C55
10N
C58
10N
C45
4.7U
C48
100N
C49
4.7U
C50
100N
CAPAS PROCHES DU FPGA
C39
100N
C41
100N
C43
100N
C46
100N
C40
10N
C42
10N
C44
10N
C47
10N
2 DANS CHAQUE COIN
1 INTERIEURE + 1 EXTERIEURE
C51
100N
C53
100N
C56
100N
C59
100N
C52
10N
C54
10N
C57
10N
C60
10N
2 DANS CHAQUE COIN
1 INTERIEURE + 1 EXTERIEURE
REF:
merger_fpga
ETUDE:
PAGE: 15/15
DESSIN:
DATE:

Annex 13 : Board implantation components (top side)

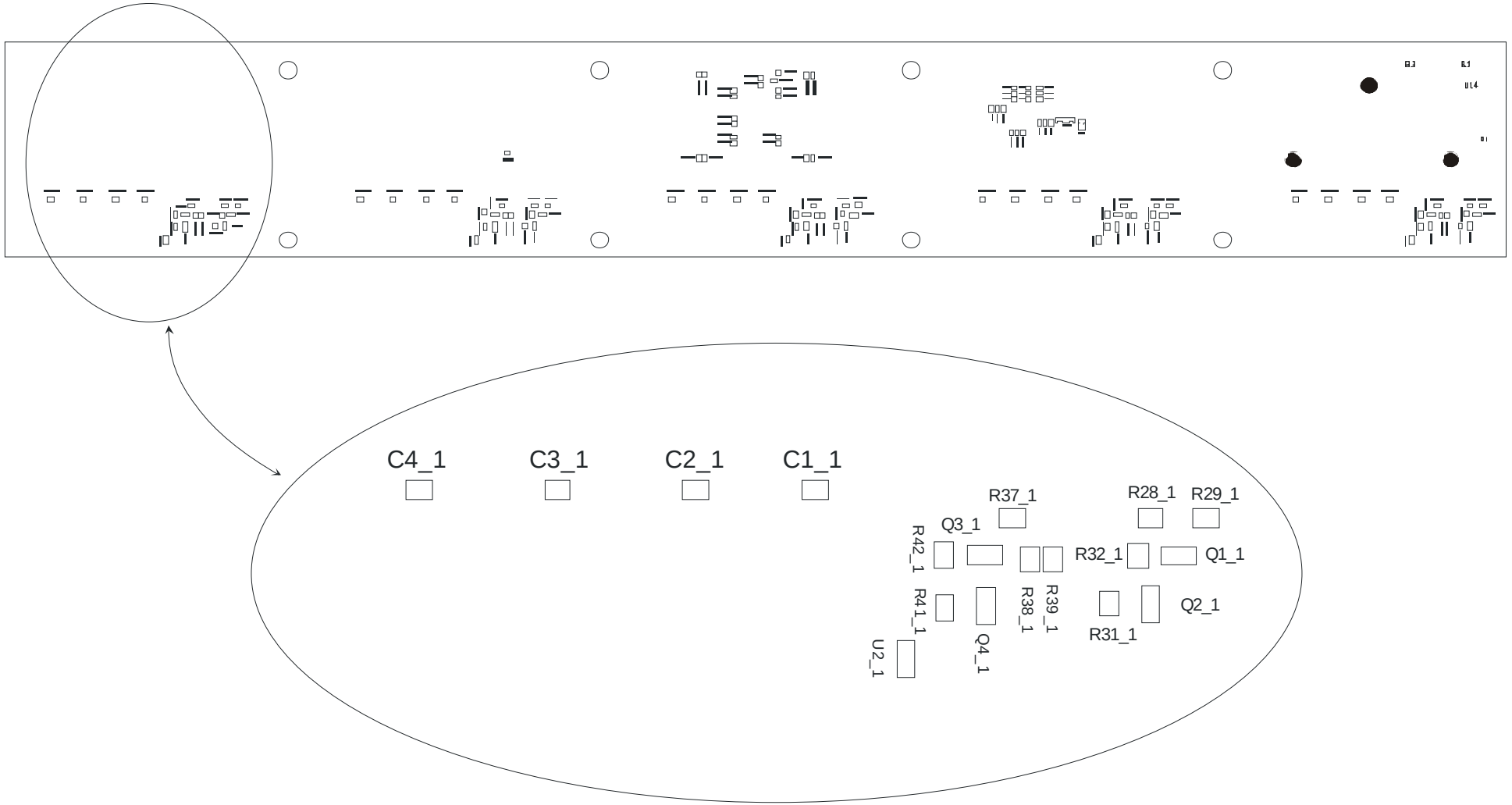


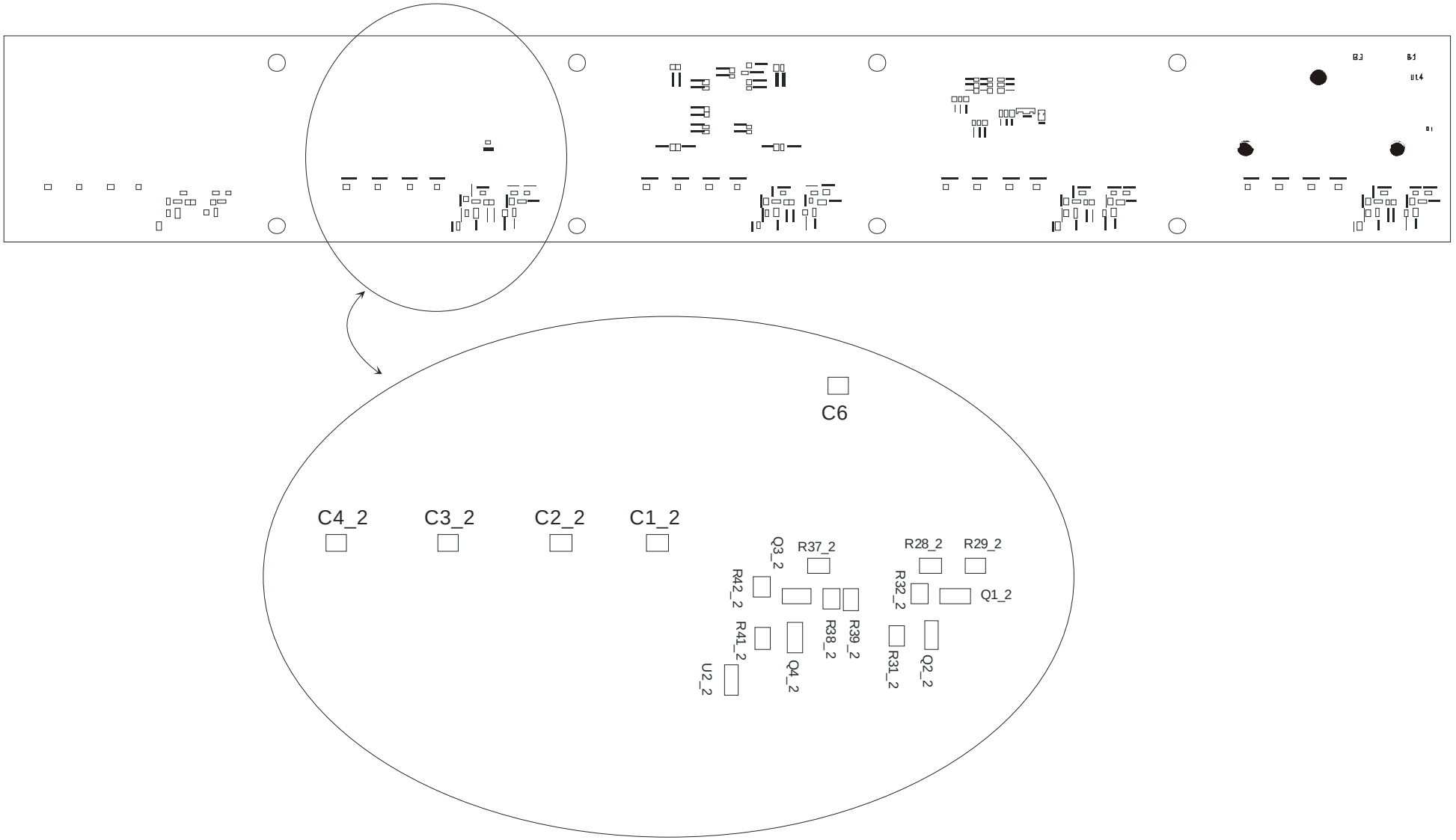

U13
U12
U13
U12
U9_4
U1_4
U4_4
U3_4
U5_4
J1_4

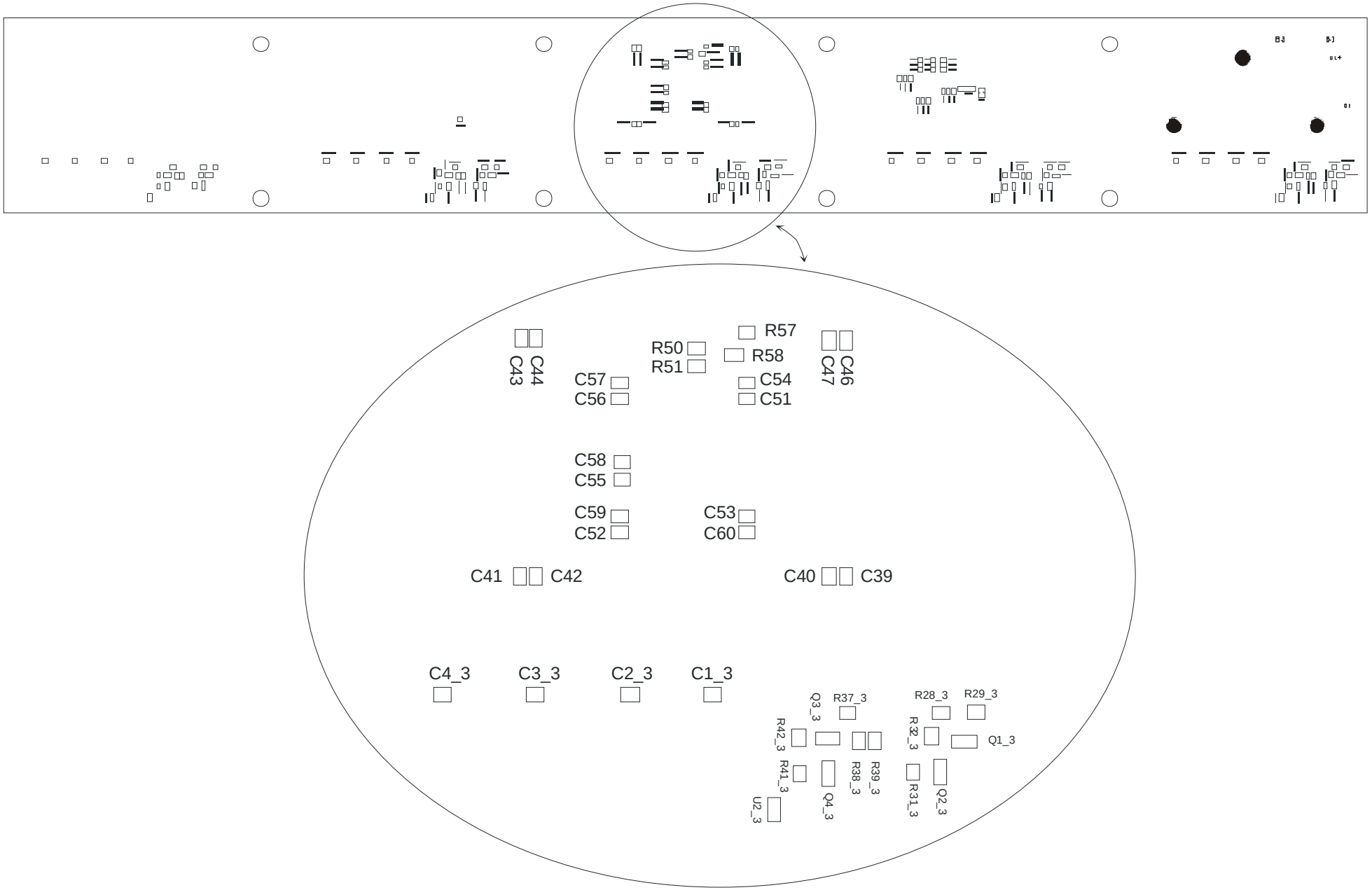

J2
P15 P12
P14 P13
P8
BP2
P7
P6 P5
P4
P1 P2 P3
P9
P10 P11
R52 R54
R53
C45 C48 C50 C49
U1
R56 R55
C12
M1
U9_3
U1_3 U4_3 U3_3 U5_3
J1_3
U13
U12

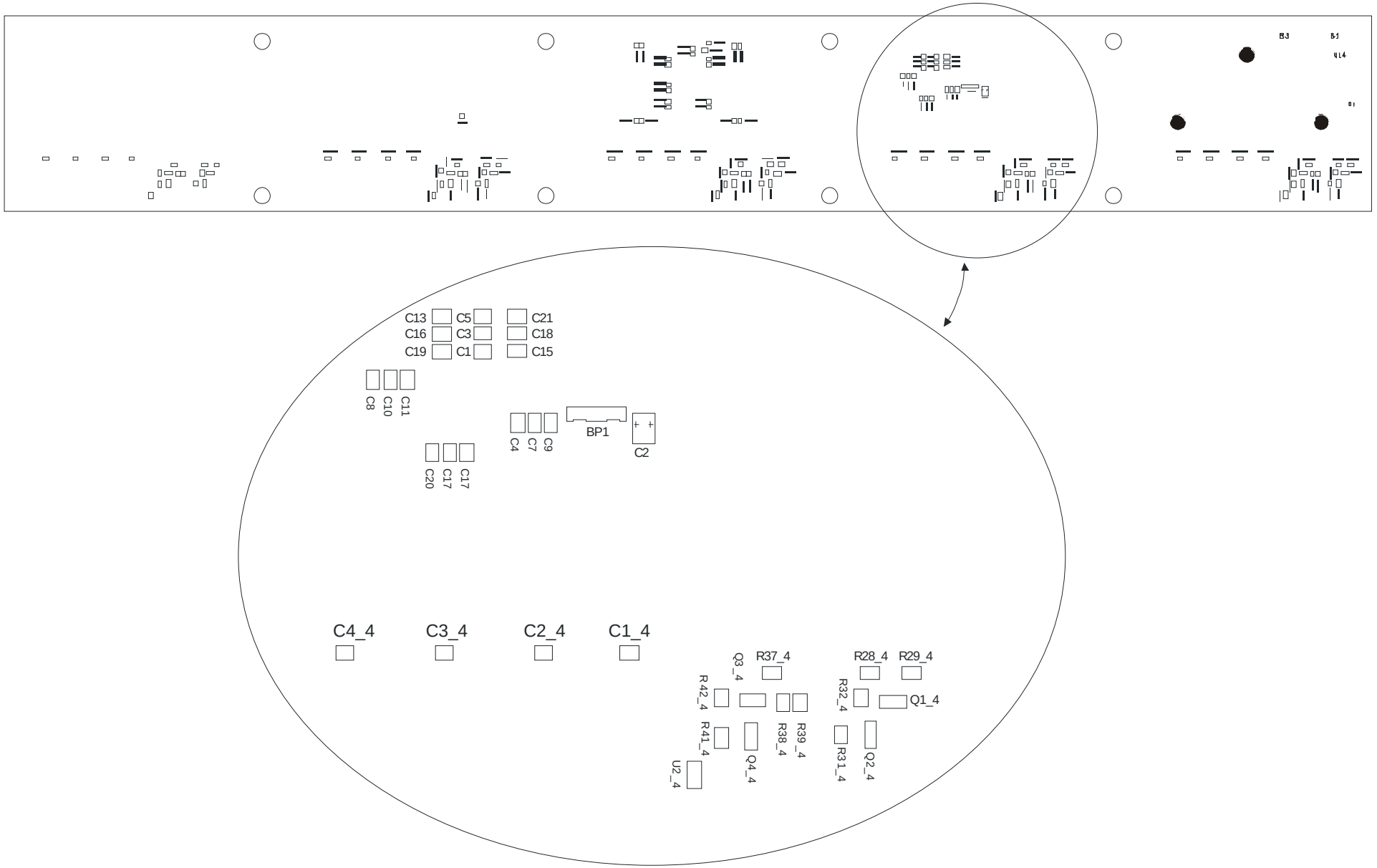

U10
U12
U13
U11
J7
B4
B2
BP3
BP4
C135
C138
R255
R260
C140
C143
U74
U75
V70
C132
C129
R261
R258
C142
C144
R244
R240
R252
R249
R242
R247
R246
R245
C133
C134
C130
C131
R239
R259
R256
C141
L12
U72
D12
R253
R254
C139
C136
C137
R257
R251
U73
R4
R1
R2
R3
U9_1
U9_2
J1_2
J1_1

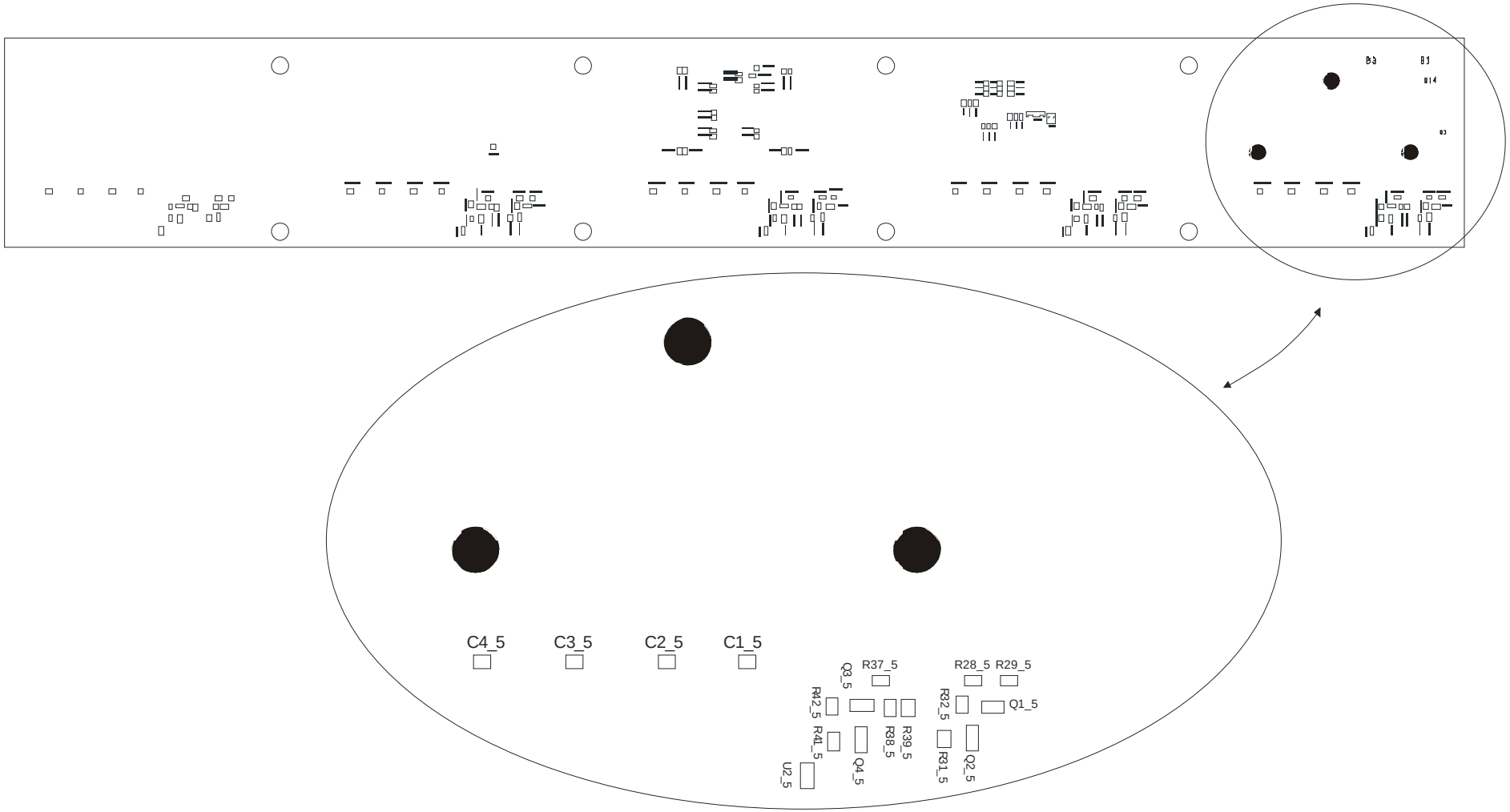
Annex 14 : Board implantation components (bottom side)




C6
C4_2
C3_2
C2_2
C1_2
Q3_2
R37_2
R28_2
R29_2
R42_2
R32_2
Q1_2
R41_2
R38_2
R39_2
R31_2
Q2_2
U2_2
Q4_2


C43
C44
R50
R51
R57
R58
C57
C56
C54
C51
C47
C46
C58
C55
C59
C52
C53
C60
C41
C42
C40
C39
C4_3
C3_3
C2_3
C1_3
Q3_3
R37_3
R28_3
R29_3
R42_3
R32_3
Q1_3
R41_3
R38_3
R39_3
U2_3
Q4_3
R31_3
Q2_3


C13
C5
C21
C16
C3
C18
C19
C1
C15
C8
C10
C11
C4
C7
C9
BP1
C2
C20
C17
C17
C4_4
C3_4
C2_4
C1_4
Q3_4
R37_4
R28_4
R29_4
R42_4
Q1_4
R32_4
R41_4
R38_4
R39_4
Q4_4
R31_4
Q2_4
U2_4


C4_5
C3_5
C2_5
C1_5
Q3_5
R37_5
R28_5
R29_5
R42_5
R41_5
U2_5
Q4_5
R38_5
R39_5
R32_5
R31_5
Q2_5
Q1_5

Annex 15 : FPGA VHDL file description

1 / FPGA VHDL file description : MERGER Function

```
--=============================================================================
--
-- Design Units : MERGER board, CREAM experiment
--
--  File name    : MERGER.vhd
--
--  Purpose      :
--
--  Notes        : MERGER FPGA Top level VHDL file
--
--  Limitations  :
--
--  Errors       :
--
--  Library      :
--
--  Dependencies :
--
--  Author       : Joel BOUVIER
--                 Laboratoire de physique Subatomique et de cosmologie
--                 53 Avenue des Martyrs
--                 38026 Grenoble Cedex, FRANCE
--
--=============================================================================
Revision List
-- Version  Author  Date      Change
-- 0.0      JB      27/01/06  Initial version
-- 0.1      JB      06/02/06  Initialiaze twac constant to 10
-- 0.2      JB      08/02/06  Change TWAC constant in std_logic_vector instead
--                            of integer
--                  17/02/06  change size of constant convert_temp ( 12 bits
--                            instead of 16 )
--                            add APA library ( for GLINT component )
--                            change value of convert_temp constant ( 334
--                            instead of 668 because the decrease loop in core
--                            •dle•wled is made in 2 state )
--                  20/02/06  Change the value of convert_temp constant to
--                            45 µs ( time adjusted before DAQ_FEE board
--                            simulation ) old value : 334h new value : 384h
--                  21/02/06  Change Convert_temp constant after CHERCAM DAQ
--                            global simulation to 3A0h
-- 0.3      JB      06/08/06  Modification in MERGER_COMMAND.vhd
--                            Modification in MERGER_CORE.vhd
--                            Add interface with the DAC and the LED
--                            Add MERGER_DAC.vhd
--=============================================================================

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
library APA;


entity merger is
  port (
     POR_RESET_G   : in  std_logic ;        -- Power on Reset, Global in, active
low
     RSTIN_N_G     : in  std_logic ;        -- hard Reset, Global In, active low
     CLK           : in  std_logic ;        -- Basis clock, Global Input

     POR_RESET     : in  std_logic ;        -- Power on Reset signal
     RSTIN_N       : in  std_logic ;        -- Power on Reset signal
     NO_ME         : in  std_logic ;        -- Merger number ( 0 or 1 )
     ENIN_N        : in  std_logic ;        -- Enable from SPARSIFICATION board
     DIN           : in  std_logic ;        -- data from SPARSIFICATION board
     TRIG_RECEIVED : out std_logic ;        -- view signal
     DAT_IN_ACK    : out std_logic ;

  SERDES link
     PD_N          : out std_logic ;        -- SERDES power down ( low active )
     CK_SERDES     : out std_logic ;        -- SERDES transmission clock
     DAT           : out std_logic_vector(20 downto 0); -- SERDES data

  DAQ_FEE link
     pw_row        : out std_logic_vector(4 downto 0);  -- command power for FLEX
     rst_row       : out std_logic_vector(4 downto 0);  -- RST signal to FLEX
     row3_row      : out std_logic_vector(4 downto 0);  -- Row3 signal to FLEX
     dout_row      : out std_logic_vector(4 downto 0);  -- data signal to DAQ_FEE
     en_n_row      : out std_logic_vector(4 downto 0);  -- Enable signal to FLEX
     sel_row       : out std_logic_vector(4 downto 0);  -- Output buffer selection
```

```
      endr_row      : out std_logic_vector(4 downto 0);  -- Input buffer selection
      dr_row0       : in  std_logic_vector(9 downto 0);  -- Input data from FLEX
      dr_row1       : in  std_logic_vector(9 downto 0);  -- Input data from FLEX
      dr_row2       : in  std_logic_vector(9 downto 0);  -- Input data from FLEX
      dr_row3       : in  std_logic_vector(9 downto 0);  -- Input data from FLEX
      dr_row4       : in  std_logic_vector(9 downto 0);  -- Input data from FLEX

    LED interface
      LED_CMD       : out std_logic ;
      DAC_SYNC_N    : out std_logic ;
      DAC_SCLK      : out std_logic ;
      DAC_DIN       : out std_logic );

 end merger ;

 architecture behave of merger is

  component merger_command
   port (
      talk_frame : in std_logic_vector(7 downto 0);
      RST        : in  std_logic ;         -- Asynchronous Reset
      CLK        : in  std_logic ;         -- Basis clock
      LD_CMD     : in  std_logic ;         -- Enable to Transmit command enable
      IDDLE      : in  std_logic ;         -- Initial state
      DIN        : in  std_logic ;         -- Data input from SPARSIFICATION
      EIN        : in  std_logic ;         -- ENABLE input from SPARSIFICATION
      DOUT       : out std_logic ;         -- DATA signal to ROW
      ENOUT      : out std_logic ;         -- Enable data signal to ROW
      ENIN       : out std_logic ;         -- ENABLE ddata signal to MERGER_CORE
module
      datin      : out std_logic_vector(15 downto 0) – Data to MERGER_CORE module
       );
  end component ;

  component merger_core
   port (
     convert_temp  : in  std_logic_vector(11 downto 0);
     trigger_frame : in  std_logic_vector(7 downto 0);
     ID_CD_MERGER  : in  std_logic_vector(3 downto 0);
     RST           : in  std_logic ;         -- Asynchronous Reset
     CLK           : in  std_logic ;         -- Basis clock
     ENIN          : in  std_logic ;         -- ENABLE data signal 2 MERGER_CORE
module
     datin         : in  std_logic_vector(7 downto 0); -- Data to MERGER_CORE
module
     BUSY          : in  std_logic ;
     NO_ME         : in  std_logic ;         -- Merger •dle• number, assigned by
switch
     LD_CMD        : out std_logic ;         -- Enable to Transmit command enable
     INDIC_CMD     : out std_logic ;         -- Command •dle•wledgement
     IDDLE         : out std_logic ;         -- Initial state
     endr          : out std_logic_vector(4 downto 0);
     sel           : out std_logic_vector(4 downto 0);
     dac_dat       : out std_logic_vector(15 downto 0);
     DAC_CMD       : out std_logic ;
     LED_CMD       : out std_logic );
  end component ;

  component merger_data_io
   port (
      twac       : in  std_logic_vector(3 downto 0) ;
      RST        : in  std_logic ;                  -- Asynchronous Reset
      CLK        : in  std_logic ;                  -- Basis clock
      NO_ME      : in  std_logic ;                  -- Merger number ( 0 or 1 )
      ENOUT      : in  std_logic ;                  -- Enable data signal to ROW
      IDDLE      : in  std_logic ;                  -- Reset STATE command
      endr       : in  std_logic_vector(4 downto 0);-- row validation
      dr_0       : in  std_logic_vector(9 downto 0);-- Data in row 0
      dr_1       : in  std_logic_vector(9 downto 0);-- Data in row 1
      dr_2       : in  std_logic_vector(9 downto 0);-- Data in row 2
      dr_3       : in  std_logic_vector(9 downto 0);-- Data in row 3
      dr_4       : in  std_logic_vector(9 downto 0);-- Data in row 4
      BUSY       : out std_logic ;
      DAT_IN_ACK : out std_logic ;
      out_data   : out std_logic_vector(20 downto 0));-- Output data
  end component ;

  component merger_power
   port (
     max_cpt_power : in std_logic_vector(15 downto 0);
     RST_N         : in  std_logic ;                  -- Asynchronous Reset
     CLK           : in  std_logic ;                  -- Basis clock
     NO_ME         : in  std_logic ;                  -- Merger number (0 or 1)
     pw_row        : out std_logic_vector(4 downto 0));-- command power for FLEX
  end component ;

  component GLINT
   port( A  : in  std_logic ;
         GL : out std_logic );
  end component;

  component merger_dac
   port (
     RST           : in  std_logic ;         -- Asynchronous Reset
```

```
    CLK           : in  std_logic ;         -- Basis clock
    DAC_CMD       : in  std_logic ;
    dac_dat       : in  std_logic_vector(15 downto 0);

    DAC_SYNC_N    : out std_logic ;
    DAC_SCLK      : out std_logic ;
    DAC_DIN       : out std_logic );
  end component ;

  signal RST_INT      : std_logic ;
  signal EIN_INT      : std_logic ;
  signal din_int      : std_logic_vector(15 downto 0);
  signal LD_CMD       : std_logic ;
  signal IDDLE        : std_logic ;
  signal BUSY         : std_logic ;
  signal DOUT_INT     : std_logic ;
  signal ENOUT_INT    : std_logic ;
  signal endr_row_int : std_logic_vector(4 downto 0);
  signal dr_row0_int  : std_logic_vector(9 downto 0);
  signal dr_row1_int  : std_logic_vector(9 downto 0);
  signal dr_row2_int  : std_logic_vector(9 downto 0);
  signal dr_row3_int  : std_logic_vector(9 downto 0);
  signal dr_row4_int  : std_logic_vector(9 downto 0);
  signal GENE_RST_INT : std_logic ;
  signal dac_dat      : std_logic_vector(15 downto 0);
  signal DAC_CMD      : std_logic ;

  constant talk_frame    : std_logic_vector(7 downto 0)  := X"54";
  constant trigger_frame : std_logic_vector(7 downto 0)  := x"50";
  constant ID_CD_MERGER  : std_logic_vector(3 downto 0)  := "1101";-- MERGER board
command identifier
  constant convert_temp  : std_logic_vector(11 downto 0) := x"3A0";-- T = 46,4 µs
  constant twac          : std_logic_vector(3 downto 0)  := "0001";
  constant max_cpt_power : std_logic_vector(15 downto 0) := x"FFFE";-- T = 1,6ms

 Begin

 M_CMD0 : merger_command
   port map ( talk_frame => talk_frame,
              RST        => RST_INT,
              CLK        => CLK,
              LD_CMD     => LD_CMD,
              IDDLE      => IDDLE,
              DIN        => DIN,
              EIN        => ENIN_N,
              DOUT       => DOUT_INT,
              ENOUT      => ENOUT_INT,
              ENIN       => EIN_INT,
              datin      => din_int);

 M_COR0 : merger_core
  port map ( convert_temp  => convert_temp,
             trigger_frame => trigger_frame,
             ID_CD_MERGER  => ID_CD_MERGER,
             RST           => RST_INT,
             CLK           => CLK,
             ENIN          => EIN_INT,
             datin         => din_int(7 downto 0),
             BUSY          => BUSY,
             NO_ME         => NO_ME,
             LD_CMD        => LD_CMD,
             INDIC_CMD     => TRIG_RECEIVED,
             IDDLE         => IDDLE,
             endr          => endr_row_int,
             sel           => sel_row ,

             dac_dat       => dac_dat,
             DAC_CMD       => DAC_CMD,
             LED_CMD       => LED_CMD );

 M_IO0 : merger_data_io
  port map ( twac       => twac,
             RST        => RST_INT,
             CLK        => CLK,
             NO_ME      => NO_ME,
             ENOUT      => ENOUT_INT ,--LD_CMD,
             IDDLE      => IDDLE,
             endr       => endr_row_int,
             dr_0       => dr_row0_int,
             dr_1       => dr_row1_int,
             dr_2       => dr_row2_int,
             dr_3       => dr_row3_int,
             dr_4       => dr_row4_int,
             BUSY       => BUSY,
             DAT_IN_ACK => DAT_IN_ACK,
             out_data   => DAT);

 M_pow : merger_power
  port map ( max_cpt_power => max_cpt_power,
             RST_N         => POR_RESET_G,
             CLK           => CLK,
             NO_ME         => NO_ME,
             pw_row        => pw_row);
```

```
 GLINT_0_inst : GLINT
  port map( A  => GENE_RST_INT ,
            GL => RST_INT);

 M_dac : merger_dac
  port map ( RST           => RST_INT,
             CLK           => CLK,
             DAC_CMD       => DAC_CMD,
             dac_dat       => dac_dat,

             DAC_SYNC_N    => DAC_SYNC_N,
             DAC_SCLK      => DAC_SCLK,
             DAC_DIN       => DAC_DIN );

Internal link

 GENE_RST_INT   <= '1' when POR_RESET_G = '0' else
                   '1' when RSTIN_N_G ='0' else
                   '0';

SERDES link

 PD_N      <= '1';
 CK_SERDES <= not CLK ;

FLEX link

 rst_row  <= RST_INT & RST_INT & RST_INT & RST_INT & RST_INT ;
 row3_row <= NO_ME & NO_ME & NO_ME & NO_ME & NO_ME ;
 dout_row <= DOUT_INT & DOUT_INT & DOUT_INT & DOUT_INT & DOUT_INT ;
 en_n_row <= ( not ENOUT_INT )
           & ( not ENOUT_INT )
           & ( not ENOUT_INT )
           & ( not ENOUT_INT )
           & ( not ENOUT_INT );
 endr_row <= endr_row_int ;

 dr_row0_int <= (not dr_row0(9)) & dr_row0(8)
              & (not dr_row0(7)) & dr_row0(6) & (not dr_row0(5)) & dr_row0(4)
              & (not dr_row0(3)) & dr_row0(2) & (not dr_row0(1)) & dr_row0(0);
 dr_row1_int <= (not dr_row1(9)) & dr_row1(8)
              & (not dr_row1(7)) & dr_row1(6) & (not dr_row1(5)) & dr_row1(4)
              & (not dr_row1(3)) & dr_row1(2) & (not dr_row1(1)) & dr_row1(0);
 dr_row2_int <= (not dr_row2(9)) & dr_row2(8)
              & (not dr_row2(7)) & dr_row2(6) & (not dr_row2(5)) & dr_row2(4)
              & (not dr_row2(3)) & dr_row2(2) & (not dr_row2(1)) & dr_row2(0);
 dr_row3_int <= (not dr_row3(9)) & dr_row3(8)
              & (not dr_row3(7)) & dr_row3(6) & (not dr_row3(5)) & dr_row3(4)
              & (not dr_row3(3)) & dr_row3(2) & (not dr_row3(1)) & dr_row3(0);
  dr_row4_int <= (not dr_row4(9)) & dr_row4(8)
              & (not dr_row4(7)) & dr_row4(6) & (not dr_row4(5)) & dr_row4(4)
              & (not dr_row4(3)) & dr_row4(2) & (not dr_row4(1)) & dr_row4(0);

 end behave;
;
```

2 / Command Sub routines

```
--=============================================================================
 --
 --  Design Units : MERGER board, CREAM experiment
 --
 --  File name    : merger_command.vhd
 --
 --  Purpose      :
 --
 --  Notes        :
 --
 --  Limitations  :
 --
 --  Errors       :
 --
 --  Library      :
 --
 --  Dependencies :
 --
 --  Author       : Joel BOUVIER
 --                 Laboratoire de physique Subatomique et de cosmologie
 --                 53 Avenue des Martyrs
 --                 38026 Grenoble Cedex, FRANCE
 --
 --=============================================================================
Revision List
 -- Version  Author  Date      Change
 -- 0.0      JB      27/01/06  Initial version
 -- 0.1      JB      06/08/06  Width shift register is 16 instead 8, increased
 --                            to avoid, on the FLEX, more clock before command
 --
 --=============================================================================

 library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

 entity merger_command is
   port (
      talk_frame : in std_logic_vector(7 downto 0);

      RST        : in  std_logic ;     -- Asynchronous Reset
      CLK        : in  std_logic ;     -- Basis clock
      LD_CMD     : in  std_logic ;     -- Enable to Transmit command enable
      IDDLE      : in  std_logic ;     -- Initial state
      DIN        : in  std_logic ;     -- Data input from SPARSIFICATION
      EIN        : in  std_logic ;     -- ENABLE input from SPARSIFICATION

      DOUT       : out std_logic ;     -- DATA signal to ROW
      ENOUT      : out std_logic ;     -- Enable data signal to ROW
      ENIN       : out std_logic ;     -- ENABLE ddata signal to MERGER_CORE module
      datin      : out std_logic_vector(15 downto 0) – Data to MERGER_CORE module
       );
end merger_command ;

architecture behave of merger_command is
  constant shift_nb   : integer := 16 ;
  signal mem_din      : std_logic_vector(1 downto 0);
  signal mem_ein      : std_logic_vector(1 downto 0);
  signal shift_din    : std_logic_vector((shift_nb – 1) downto 0);
  signal shift_ein    : std_logic_vector((shift_nb – 1) downto 0);

Begin

 mem_signal : process ( RST, CLK )
   begin
     if RST ='1' then
       mem_din <= ( others =>'0');
       mem_ein <= ( others =>'0');
     elsif rising_edge(CLK) then
       mem_din <= mem_din(0) & DIN ;
       mem_ein <= mem_ein(0) & ( not EIN );
     end if;
   end process mem_signal;

 shift_signal : process ( RST, CLK )
   begin
     if RST ='1' then
       shift_din <= ( others => '0');
       shift_ein <= ( others => '0');
     elsif rising_edge(CLK) then
       if LD_CMD = '1' then
         shift_din((shift_nb – 1) downto 8) <= ( others =>'0');
         shift_din(7 downto 0)              <= talk_frame;
         shift_ein((shift_nb – 1) downto 8) <= ( others =>'0');
         shift_ein(7 downto 0)              <= x"FF";
       else
         shift_din <= shift_din((shift_nb – 2) downto 0) & mem_din(1);
```

```
          shift_ein <= shift_ein((shift_nb – 2) downto 0) & mem_ein(1);
        end if;
      end if;
    end process shift_signal;

    ENIN  <= mem_ein(1);
    datin <= shift_din ;
    DOUT  <= shift_din((shift_nb – 1));
    ENOUT <= shift_ein((shift_nb – 1));

end behave;;
```

3 / CORE Subroutines

```
--=============================================================================
 --
 --  Design Units : MERGER board, CREAM experiment
 --
 --  File name    : merger_core.vhd
 --
 --  Purpose      :
 --
 --  Notes        :
 --
 --  Limitations  :
 --
 --  Errors       :
 --
 --  Library      :
 --
 --  Dependencies :
 --
 --  Author       : Joel BOUVIER
 --                 Laboratoire de physique Subatomique et de cosmologie
 --                 53 Avenue des Martyrs
 --                 38026 Grenoble Cedex, FRANCE
 --
 --=============================================================================
 Revision List
 -- Version  Author  Date      Change
 -- 0.0      JB      27/01/06    Initial version
 -- 0.2      JB      08/01/06    Change the number of clock_row after the
 --                               enable_row ( initial value = 7 ( 3 CLK )
 --                                          new value     = 15 ( 17 CLK ))
 --                  17/02/06    change size of constant convert_temp ( 12 bits
 --                               instead of 16 )
 -- 0.3      JB      06/08/06    Now the buffer driver to ROW is not valided at
 --                               RESET ( Enable, CLk, Data to ROW is high state )
 --                               The clock number after rising /enable is set
 --                               to 48. ( request for DAQFEE board, August 2006 )
 --                               Add 48 CL after reading a ROW and before
 --                               select the next ROW ( request for DAQFEE board,
 --                               August 2006 )
 --=============================================================================

 library ieee;
 use ieee.std_logic_1164.all;
 use ieee.std_logic_arith.all;
 use ieee.std_logic_unsigned.all;

 entity merger_core is
   port (
     convert_temp  : in  std_logic_vector(11 downto 0);
     trigger_frame : in  std_logic_vector(7 downto 0);
     ID_CD_MERGER  : in  std_logic_vector(3 downto 0);

     RST           : in  std_logic ;     -- Asynchronous Reset
     CLK           : in  std_logic ;     -- Basis clock
     ENIN          : in  std_logic ;     -- ENABLE data signal 2 MERGER_CORE module
     datin         : in  std_logic_vector(7 downto 0); --Data to MERGER_CORE module
     BUSY          : in  std_logic ;
     NO_ME         : in  std_logic ;     -- Merger •dle• number, assigned by switch

     LD_CMD        : out std_logic ;     -- Enable to Transmit command enable
     INDIC_CMD     : out std_logic ;     -- Command •dle•wledgement
     IDDLE         : out std_logic ;     -- Initial state
     endr          : out std_logic_vector(4 downto 0);
     sel           : out std_logic_vector(4 downto 0);

     dac_dat       : out std_logic_vector(15 downto 0);
     DAC_CMD       : out std_logic ;
     LED_CMD       : out std_logic
     );
 end merger_core ;

 architecture behave of merger_core is
   signal endr_int     : std_logic_vector(4 downto 0);
   signal sel_int      : std_logic_vector(4 downto 0);

   signal cmd_int      : std_logic_vector(7 downto 0);
   signal cpt_wconvert : std_logic_vector(11 downto 0);
   type list_basis_state is ( fsm_bas0, fsm_bas1, fsm_bas2 );
   signal basis_state       : list_basis_state;
   type list_core_state is ( idle,
                             cmd_wat0, cmd_wat1, cmd_wat2,
                             cmd_ack0, cmd_ack1, cmd_ack2, cmd_ack3, cmd_ack4,
                             cmd_ack5, cmd_ack6, cmd_ack7, cmd_ack8, cmd_ack9,
                             cmd_lec0, cmd_lec1, cmd_lec2, cmd_lec3, cmd_lec4,
                             cmd_lec5, cmd_lec6, cmd_lec7, cmd_lec8, cmd_lec9,
                             cmd_lec10, cmd_lec11,
                             cmd_dela0, cmd_dela1, cmd_dela2, cmd_dela3);
   signal core_state       : list_core_state;
```

```
  signal cpt_ena_ack   : std_logic_vector(2 downto 0);
  signal INDIC_CMD_INT : std_logic ;
  signal cpt_cmd_ack   : std_logic_vector(3 downto 0);
  signal DAC_VAL       : std_logic ;
  signal LED_CMD_INT   : std_logic_vector(6 downto 0);

Begin

  main_process : process ( RST, CLK )
  begin
    if RST ='1' then
      INDIC_CMD_INT   <= '0';
      cpt_ena_ack <= ( others => '0' );
      basis_state <= fsm_bas0 ;
    elsif rising_edge(CLK) then
      case basis_state is
        when fsm_bas0 => if ENIN = '1' then
                           basis_state <= fsm_bas1 ;
                           INDIC_CMD_INT   <= '1';
                           cpt_ena_ack <= ( others => '0' );
                         end if ;
        when fsm_bas1 => if cpt_ena_ack = "111" then
                           basis_state <= fsm_bas2 ;
                           INDIC_CMD_INT   <= '0';
                         else
                           cpt_ena_ack <= cpt_ena_ack + 1;
                         end if;
        when fsm_bas2 => if ENIN = '0' then
                           basis_state <= fsm_bas0 ;
                         end if ;
      end case ;
    end if ;
  end process;

  slave_process : process ( RST, CLK )
  begin
    if RST ='1' then
      sel_int        <= ( others =>'0'); -- At start up the clock is not generated
      endr_int       <= ( others =>'0'); -- on all FLEX
      cpt_cmd_ack    <= ( others =>'0');
      cmd_int        <= ( others =>'0');
      cpt_wconvert   <= ( others =>'0');
      LD_CMD         <= '0';
      core_state     <= idle ;
      dac_dat        <= ( others =>'0');
      DAC_VAL        <= '0';
      DAC_CMD        <= '0';
      LED_CMD_INT(0) <= '0';
    elsif rising_edge(CLK) then
      case core_state is
        when idle      => DAC_CMD             <= '0';
                          if INDIC_CMD_INT = '1' then
                            core_state  <= cmd_wat0;
                          end if;

        when cmd_wat0  => core_state  <= cmd_wat1;
        when cmd_wat1  => core_state  <= cmd_wat2;
        when cmd_wat2  => core_state  <= cmd_ack0;
                          cpt_cmd_ack <= ( others =>'0');
                          if datin(3 downto 0) /= ID_CD_MERGER then
                            sel_int    <= ( others =>'1');
                          end if;

        when cmd_ack0  => if cpt_cmd_ack(1 downto 0) = "11" then
                            core_state <= cmd_ack1 ;
                            cmd_int    <= datin ;
                          else
                            core_state  <= cmd_ack0 ;
                            cpt_cmd_ack <= cpt_cmd_ack + 1 ;
                          end if;
        when cmd_ack1  => if cmd_int = trigger_frame then
                            core_state     <= cmd_lec0 ;
                            LED_CMD_INT(0) <= DAC_VAL;
                          else
                            core_state <= cmd_ack2 ;
                            cpt_cmd_ack <= ( others =>'0');
                          end if;

Not a TRIGGER Command

        when cmd_ack2  => if cpt_cmd_ack(1 downto 0) = "11" then
                            case cmd_int is
                              when x"D0"  => core_state           <= idle ;
                                             dac_dat(7 downto 0)  <= datin;
                              when x"D1"  => core_state           <= idle ;
                                             dac_dat(15 downto 8) <= datin;
                                             DAC_CMD              <= '1';
                              when x"D2"  => core_state <= idle ;
                                             DAC_VAL    <= '1';
                              when x"D3"  => core_state <= idle ;
                                             DAC_VAL    <= '0';
                              when others => core_state <= cmd_ack4 ;
                            end case ;
```

```
                    else
                      core_state <= cmd_ack3 ;
                    end if;
        when cmd_ack3  => core_state  <= cmd_ack2 ;
                    cpt_cmd_ack <= cpt_cmd_ack + 1;
        when cmd_ack4  => core_state  <= cmd_ack5 ;
                    cpt_cmd_ack <= ( others =>'0');
        when cmd_ack5  => if cpt_cmd_ack = "1111" then
                      core_state <= cmd_ack9 ;
                    else
                      core_state <= cmd_ack6 ;
                    end if;
        when cmd_ack6  => core_state  <= cmd_ack7 ;
                    cpt_cmd_ack <= cpt_cmd_ack + 1;
        when cmd_ack7  => core_state  <= cmd_ack8 ;
        when cmd_ack8  => core_state  <= cmd_ack5 ;

        when cmd_ack9  => core_state <= idle ;
                    sel_int <= ( others =>'0');

Processing TRIGGER command

        when cmd_lec0  => core_state     <= cmd_lec1 ;
                    cpt_wconvert   <= ( others =>'0');
                    LED_CMD_INT(0) <= '0';
        when cmd_lec1  => if cpt_wconvert = convert_temp then
                      core_state <= cmd_lec3 ;
                    else
                      core_state <= cmd_lec2 ;
                    end if;
        when cmd_lec2  => core_state   <= cmd_lec1;
                    cpt_wconvert <= cpt_wconvert + 1 ;
        when cmd_lec3  => core_state <= cmd_lec4 ;
                    sel_int  <= "00001";
                    endr_int <= "00001";
        when cmd_lec4  => if NO_ME ='0' then
                      core_state <= cmd_lec5;
                    else
                      core_state  <= cmd_lec11 ;
                      cpt_cmd_ack <= ( others =>'0');
                    end if;
        when cmd_lec5  => core_state <= cmd_lec6 ;
                    LD_CMD <= '1';
        when cmd_lec6  => core_state <= cmd_lec7 ;
                    LD_CMD <= '0';
        when cmd_lec7  => if BUSY = '1' then
                      core_state <= cmd_lec8 ;
                    end if;
        when cmd_lec8  => if BUSY = '0' then
                      core_state <= cmd_dela0 ;
                    end if;

wait at end of the read command : 48 CLK must be valid
( DAQFEE requirement : see B.Boyer august 2006 )

        when cmd_dela0 => core_state  <= cmd_dela1;
                    cpt_cmd_ack <= ( others =>'0');
        when cmd_dela1 => if cpt_cmd_ack = "1111" then
                      core_state <= cmd_lec9 ;
                    else
                      core_state <= cmd_dela2 ;
                    end if;
        when cmd_dela2 => core_state  <= cmd_dela3 ;
                    cpt_cmd_ack <= cpt_cmd_ack + 1;
        when cmd_dela3 => core_state  <= cmd_dela1 ;

Go to the next FLEX or END if all FLEX are read

        when cmd_lec9  => core_state <= cmd_lec10;
                    sel_int  <= sel_int(3 downto 0) & '0';
                    endr_int <= endr_int(3 downto 0) & '0';
        when cmd_lec10 => if sel_int = "00000" then
                      core_state <= idle ;
                    else
                      core_state <= cmd_lec5;
                    end if;
        when cmd_lec11 => if cpt_cmd_ack(2 downto 0) = "111" then
                      core_state  <= cmd_lec5;
                    else
                      core_state  <= cmd_lec11;
                      cpt_cmd_ack <= cpt_cmd_ack + 1 ;
                    end if;
      end case;
    end if;
  end process ;

  process ( RST, CLK )
  begin
    if RST ='1' then
      LED_CMD_INT(6 downto 1) <= ( others => '0' );
    elsif rising_edge(CLK) then
      LED_CMD_INT(6 downto 1) <= LED_CMD_INT(5 downto 0);
    end if ;
  end process;
```

```
  INDIC_CMD <= INDIC_CMD_INT;
  endr      <= endr_int ;
  sel       <= sel_int ;
  IDDLE     <= '0';

  LED_CMD <= LED_CMD_INT(6) ;

end behave;
```

4 / MERGER_DATA_IO Subroutines

```
--=============================================================================
--
--  Design Units : MERGER board, CREAM experiment
--
--  File name    : merger_din_io.vhd
--
--  Purpose      :
--
--  Notes        :
--
--  Limitations  :
--
--  Errors       :
--
--  Library      :
--
--  Dependencies :
--
--  Author       : Joel BOUVIER
--                 Laboratoire de physique Subatomique et de cosmologie
--                 53 Avenue des Martyrs
--                 38026 Grenoble Cedex, FRANCE
--
--=============================================================================
Revision List
-- Version  Author  Date      Change
-- 0.0      JB      16/03/05  Initial version
-- 0.1      JB      06/02/06  Correct error in row number generation in out data
( row number is active only for 151 values
instead 160 )
-- 0.2      JB      08/02/06  Remove Clock Enable for the DATin DATout signal
generation, replace by a if condition
Change twac in std_logic_vector instead of integer
--
--=============================================================================

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity merger_data_io is
  port (
     twac       : in  std_logic_vector(3 downto 0);
     RST        : in  std_logic ;                   -- Asynchronous Reset
     CLK        : in  std_logic ;                   -- Basis clock
     NO_ME      : in  std_logic ;                   -- Merger number ( 0 or 1 )
     ENOUT      : in  std_logic ;                   -- Enable data signal to ROW
     IDDLE      : in  std_logic ;                   -- Reset STATE command
     endr       : in  std_logic_vector(4 downto 0); -- row validation
     dr_0       : in  std_logic_vector(9 downto 0); -- Data in row 0
     dr_1       : in  std_logic_vector(9 downto 0); -- Data in row 1
     dr_2       : in  std_logic_vector(9 downto 0); -- Data in row 2
     dr_3       : in  std_logic_vector(9 downto 0); -- Data in row 3
     dr_4       : in  std_logic_vector(9 downto 0);  -- Data in row 4

     BUSY       : out std_logic ;
     DAT_IN_ACK : out std_logic ;
     out_data   : out std_logic_vector(20 downto 0) ); -- Output data

end merger_data_io ;

architecture behave of merger_data_io is
  type data_arch is array (0 to 9 ) of std_logic_vector(17 downto 0);
  signal datin    : data_arch ;
  signal datout   : data_arch ;
  signal mdat_int : std_logic_vector(9 downto 0);
  signal cdat_int : std_logic_vector(9 downto 0);

 State machine in MERGER_FSM_IO sub module

 type liste_etat is ( idle, ack_cmd, wait_twac,
                      daq_bit0, daq_bit1, daq_bit2, daq_bit3,
                      daq_bit4, daq_bit5, daq_bit6, daq_bit7,
                      daq_word0, daq_word1, daq_word2, daq_word3, daq_word4,
                      daq_end );
  signal etat       : liste_etat;
  signal endr_int   : std_logic_vector(4 downto 0);  -- row validation memorization

 State machine in MERGER_SHIFT_IO sub module

  type l_shift_io_state is ( SHIFT_IO_BEGIN, SHIFT_IO_ACK,  SHIFT_IO_INIT,
                             SHIFT_IO_INIT0, SHIFT_IO_INIT1,  SHIFT_IO_INIT2,
                             SHIFT_IO_INIT3, SHIFT_IO_INIT4,  SHIFT_IO_INIT5,
                             SHIFT_IO_INIT6, SHIFT_IO_INIT7,  SHIFT_IO_INIT8,
                             SHIFT_IO_INIT9, SHIFT_IO_INIT10, SHIFT_IO_INIT11);
  signal shift_io_state : l_shift_io_state;
  signal CDAT       : std_logic ;
```

```
  signal cpt_cdat   : std_logic_vector(3 downto 0);
  signal column_ad  : std_logic_vector(3 downto 0);
  signal row_ad     : std_logic_vector(2 downto 0);
  signal row_ad_int : std_logic_vector(2 downto 0);
  signal LOW_HIGH   : std_logic ;
  signal V_DAT_INT  : std_logic ;

  signal ld_bit     : std_logic_vector(9 downto 0);  -- load data row
  signal LOAD_BIT   : std_logic ;
  signal LOAD_WORD  : std_logic ;
  signal MDAT       : std_logic ;
  signal cpt_bit    : std_logic_vector(4 downto 0);  -- word length : 18 bit
  signal cpt_word   : std_logic_vector(3 downto 0);  -- ASIC length : 16 word
  signal cpt_twac   : std_logic_vector(3 downto 0);  -- counter for Twac
Begin
```

MERGER_FSM_IO subsystem

```
process ( RST, CLK )
begin
  if RST ='1' then
    etat      <= idle ;
    LOAD_BIT  <= '0';
    load_word <= '0';
    cpt_bit   <= ( others => '0');
    cpt_word  <= ( others => '0');
    cpt_twac  <= ( others => '0');
    busy      <= '0';
    endr_int  <= ( others => '0');
  elsif rising_edge(CLK) then
    case etat is
       when idle      => if ENOUT ='1' and endr /= "00000" then
                           etat <= ack_cmd ;
                           BUSY <= '1';
                           endr_int  <= endr ;
                         end if;
       when ack_cmd   => if ENOUT = '0' then
                           etat     <= wait_twac ;
                           cpt_twac <= twac - 1;
                         end if;
       when wait_twac => if cpt_twac /= "0000" then
                           cpt_twac <= cpt_twac - 1 ;
                           etat     <= wait_twac ;
                         else
                           etat     <= daq_bit0 ;
                         end if ;
       when daq_bit0  => etat     <= daq_bit1 ;
       when daq_bit1  => etat     <= daq_bit2 ;
       when daq_bit2  => etat     <= daq_bit3 ;
                         LOAD_BIT <= '1' ;
       when daq_bit3  => etat     <= daq_bit4 ;
                         LOAD_BIT <= '0' ;
       when daq_bit4  => if cpt_bit = "10001" then
                           etat <= daq_word0 ;
                         else
                           etat <= daq_bit5 ;
                         end if ;
       when daq_bit5  => etat      <= daq_bit6 ;
                         cpt_bit   <= cpt_bit + 1 ;
       when daq_bit6  => etat      <= daq_bit7 ;
       when daq_bit7  => etat      <= daq_bit0 ;
       when daq_word0 => etat      <= daq_word1 ;
                         cpt_bit   <= ( others => '0');
       when daq_word1 => etat      <= daq_word2 ;
                         load_word <= '1';
       when daq_word2 => etat      <= daq_word3 ;
                         load_word <= '0';
       when daq_word3 => if cpt_word /= "1111" then
                           etat <= daq_word4 ;
                         else
                           etat <= daq_end ;
                         end if;
       when daq_word4 => etat     <= daq_bit2 ;
                         cpt_word <= cpt_word + 1 ;
       when daq_end   => etat     <= idle ;
                         cpt_word <= ( others => '0' );
                         BUSY     <= '0';
    end case ;
  end if;
end process ;
```

MERGER_DIN_IO subsystem

```
merger_din_io : for I in 0 to 9 generate
  process ( RST, CLK, ld_bit )
  begin
    if RST ='1' then
      datin(i) <= ( others => '0');
    elsif rising_edge( CLK ) then
      if ld_bit(i) ='1' then
        case endr is
          when "00001" => datin(i)(0) <= dr_0(i) ;
          when "00010" => datin(i)(0) <= dr_1(i) ;
          when "00100" => datin(i)(0) <= dr_2(i) ;
```

```
          when “01000” => datin(i)(0) <= dr_3(i) ;
          when “10000” => datin(i)(0) <= dr_4(i) ;
          when others  => datin(i)(0) <= datin(i)(0) ;
        end case ;
        datin(i)(17 downto 1) <= datin(i)(16 downto 0);
      else
        datin(i) <= datin(i) ;
      end if;
    end if;
  end process ;
end generate;
```

MERGER_DOUT_IO subsystem

```
merger_dout_io : for k in 0 to 9 generate

  process ( RST, CLK )
  begin
    if RST =’1’ then
      mdat_int(k) <= ‘0’;
      cdat_int(k) <= ‘0’;
    elsif rising_edge(CLK) then
      mdat_int(k) <= mdat ;
      cdat_int(k) <= cdat ;
    end if;
  end process ;

datout_int : process ( RST, CLK, cdat_int )
  begin
    if RST =’1’ then
      datout(k)(17 downto 0) <= ( others =>’0’);
    elsif rising_edge(CLK) then
      if  cdat_int(k) =’1’ then
        if mdat_int(k) =’1’ then
          datout(k) <= datin(k) ;
        else
          if k = 9 then
            datout(k) <= ( others =>’0’);
          else
            datout(k) <= datout(k+1);
          end if;
        end if;
      end if;
    end if ;
  end process datout_int;
```

```
end generate;

out_data_gene : process ( RST,CLK)
  begin
    if RST=’1’ then
      out_data(20 downto 19) <= “11”;
      out_data(18 downto 0)  <= ( others => ‘0’);
    elsif rising_edge(CLK) then
      case LOW_HIGH is
        when ‘0’    => out_data(20)           <= not V_DAT_INT ;
                       out_data(19)           <= LOW_HIGH ;
                       out_data(18 downto 14) <= “00000”;
                       out_data(13)           <= datout(0)(13);  -- FAULT bit
                       out_data(12)           <= datout(0)(12);  -- Gain value bit
                       out_data(11 downto 8)  <= column_ad; -- Column number
                       out_data(7 downto 4)   <= NO_ME & row_ad ; -- ROW number
                       out_data(3 downto 0)   <= datout(0)(17 downto 14); -- channel
number
        when others => out_data(20)           <= not V_DAT_INT ;
                       out_data(19)           <= LOW_HIGH ;
                       out_data(18 downto 12) <= “0000000”;
                       out_data(11 downto 0)  <= datout(0)(11 downto 0); -- ADC value
      end case;
    end if;
  end process out_data_gene;
```

MERGER_SHIFT_IO subsystem

```
merger_shift_io : for j in 0 to 9 generate
  process ( RST, CLK )
  begin
    if RST =’1’ then
      ld_bit(j)   <= ‘0’;
    elsif rising_edge(CLK) then
      ld_bit(j)   <= LOAD_BIT ;
    end if;
  end process ;
end generate merger_shift_io ;

MDAT_GENE : process ( RST, CLK )
  begin
    if RST =’1’ then
      MDAT <= ‘0’;
    elsif rising_edge(CLK) then
      MDAT <= load_word ;
    end if;
  end process MDAT_GENE;
```

```
DAT_IN_ACK_GENE : process ( RST, CLK )
  begin
    if RST ='1' then
      DAT_IN_ACK <= '0';
    elsif rising_edge(CLK) then
      DAT_IN_ACK <= LOAD_BIT ;
    end if;
  end process DAT_IN_ACK_GENE ;

  process ( RST, CLK )
  begin
    if RST ='1' then
      CDAT       <= '0';
      cpt_cdat   <= ( others =>'0');
      column_ad  <= ( others =>'0');
      row_ad     <= ( others =>'0');
      LOW_HIGH   <= '1';
      V_DAT_INT  <= '0';
      shift_io_state <= SHIFT_IO_BEGIN;
    elsif rising_edge(CLK) then
      case shift_io_state is
        when SHIFT_IO_BEGIN  => shift_io_state <= SHIFT_IO_ACK;
                                CDAT       <= '0';
                                cpt_cdat   <= ( others =>'0');
        when SHIFT_IO_ACK    => if LOAD_WORD ='1' then
                                  shift_io_state <= SHIFT_IO_INIT;
                                  CDAT     <= '1';
                                  row_ad     <= row_ad_int;
                                end if;

        when SHIFT_IO_INIT   => shift_io_state <= SHIFT_IO_INIT0;
                                CDAT       <= '0';
                                column_ad  <= cpt_cdat;
        when SHIFT_IO_INIT0  => shift_io_state <= SHIFT_IO_INIT1;
                                LOW_HIGH   <='0';
        when SHIFT_IO_INIT1  => if cpt_cdat = "0100" then
                                  shift_io_state <= SHIFT_IO_INIT2;
                                else
                                  shift_io_state <= SHIFT_IO_INIT3;
                                end if;
        when SHIFT_IO_INIT2  => shift_io_state <= SHIFT_IO_INIT4;
                                cpt_cdat <= "0111";
                                V_DAT_INT  <='1';
        when SHIFT_IO_INIT3  => shift_io_state <= SHIFT_IO_INIT4;
                                V_DAT_INT  <='1';
        when SHIFT_IO_INIT4  => shift_io_state <= SHIFT_IO_INIT5;
        when SHIFT_IO_INIT5  => shift_io_state <= SHIFT_IO_INIT6;
                                V_DAT_INT  <='0';
        when SHIFT_IO_INIT6  => shift_io_state <= SHIFT_IO_INIT7;
        when SHIFT_IO_INIT7  => LOW_HIGH   <='1';
                                shift_io_state <= SHIFT_IO_INIT8;
        when SHIFT_IO_INIT8  => if IDDLE = '0' then
                                  shift_io_state <= SHIFT_IO_INIT9;
                                else
                                  shift_io_state <= SHIFT_IO_BEGIN;
                                end if;
        when SHIFT_IO_INIT9  => V_DAT_INT      <='1';
                                shift_io_state <= SHIFT_IO_INIT10;
        when SHIFT_IO_INIT10 => shift_io_state <= SHIFT_IO_INIT11;
                                cpt_cdat       <= cpt_cdat + 1 ;
        when SHIFT_IO_INIT11 => CDAT           <= '1';
                                V_DAT_INT      <= '0';
                                if cpt_cdat = "1101" then
                                  shift_io_state <= SHIFT_IO_BEGIN;
                                else
                                  shift_io_state <= SHIFT_IO_INIT;
                                end if;
      end case;
    end if;
  end process;

  row_ad_int <= "000" when endr_int = "00001" else
                "001" when endr_int = "00010" else
                "010" when endr_int = "00100" else
                "011" when endr_int = "01000" else
                "100";

end behave;
```

5 / Power control Sub routines

```
--=============================================================================
--
--  Design Units : MERGER board, CREAM experiment
--
--  File name    : merger_power.vhd
--
--  Purpose      :
--
--  Notes        :
--
--  Limitations  :
--
--  Errors       :
--
--  Library      :
--
--  Dependencies :
--
--  Author       : Joel BOUVIER
--                 Laboratoire de physique Subatomique et de cosmologie
--                 53 Avenue des Martyrs
--                 38026 Grenoble Cedex, FRANCE
--
--=============================================================================
Revision List
-- Version  Author  Date      Change
-- 0.0      JB      29/01/06  Initial version
--
--=============================================================================

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity merger_power is
  port (
    max_cpt_power : in std_logic_vector(15 downto 0);
    RST_N         : in  std_logic ;                  -- Asynchronous Reset
    CLK           : in  std_logic ;                  -- Basis clock
    NO_ME         : in  std_logic ;                  -- Merger number (0 or 1)

    pw_row : out std_logic_vector(4 downto 0)); -- command power for FLEX
end merger_power ;

architecture behave of merger_power is
  signal cpt_power       : std_logic_vector(15 downto 0);
  signal CARRY_CPT_POWER : std_logic ;
  signal pw_row_int      : std_logic_vector(9 downto 0);
Begin

 POWER : process ( RST_N, CLK )
   begin
     if RST_N ='0' then
       pw_row_int      <= ( others =>'0');
       cpt_power       <= ( others =>'0');
       CARRY_CPT_POWER <= '0';
     elsif rising_edge(CLK) then

       if cpt_power = max_cpt_power then
         cpt_power       <= ( others =>'0');
         CARRY_CPT_POWER <= '1';
       else
         cpt_power       <= cpt_power + 1;
         CARRY_CPT_POWER <= '0';
       end if ;

       if CARRY_CPT_POWER = '1' then
         case NO_ME is
           when '0'    => pw_row_int <= pw_row_int(8 downto 5) & "111111";
           when others => pw_row_int <= pw_row_int(8 downto 0) & '1';
         end case ;
       end if ;
     end if;
   end process ;

 pw_row <= pw_row_int( 9 downto 5) ;

end behave;
```

6 / DAC interface Sub routines

```
--================================================================================
 --
 --  Design Units : MERGER board, CREAM experiment
 --
 --  File name    : merger_dac.vhd
 --
 --  Purpose      :
 --  Notes        :
 --  Limitations  :
 --  Errors       :
 --  Library      :
 --  Dependencies :
 --  Author       : Joel BOUVIER
 --                 Laboratoire de physique Subatomique et de cosmologie
 --                 53 Avenue des Martyrs
 --                 38026 Grenoble Cedex, FRANCE
 --
 --================================================================================
 Revision List
 -- Version  Author  Date      Change
 -- 0.0      JB      07/08/06    Initial version
 --================================================================================

 library ieee;
 use ieee.std_logic_1164.all;
 use ieee.std_logic_arith.all;
 use ieee.std_logic_unsigned.all;

 entity merger_dac is
   port (
     RST           : in  std_logic ;         -- Asynchronous Reset
     CLK           : in  std_logic ;         -- Basis clock
     DAC_CMD       : in  std_logic ;
     dac_dat       : in  std_logic_vector(15 downto 0);

     DAC_SYNC_N    : out std_logic ;
     DAC_SCLK      : out std_logic ;
     DAC_DIN       : out std_logic );
 end merger_dac ;

 architecture behave of merger_dac is
   signal data_decal : std_logic_vector(16 downto 0);
   type list_dac_state is ( •dle, sub_seq0, sub_seq1, sub_seq2, sub_seq3 );
   signal dac_state  : list_dac_state;
   signal cpt_seq    : std_logic_vector(4 downto 0);
 Begin

   process ( RST, CLK )
     begin
       if RST ='1' then
         DAC_SYNC_N <= '1';
         DAC_SCLK   <= '1';
         cpt_seq    <= ( others =>'0');
         data_decal <= ( others =>'0');
         dac_state  <= •dle ;
       elsif rising_edge(CLK) then
         case dac_state is
           when •dle    => DAC_SCLK   <= '1';
                           DAC_SYNC_N <= '1';
                           if DAC_CMD = '1' then
                             dac_state  <= sub_seq0 ;
                             cpt_seq    <= ( others =>'0');
                             data_decal <= '0' & "00" & dac_dat(13 downto 0) ;
                           end if;
           when sub_seq0 => dac_state  <= sub_seq1 ;
                           data_decal <= data_decal(15 downto 0) & '0';
                           DAC_SYNC_N <= '0';
                           DAC_SCLK   <= '1';
           when sub_seq1 => dac_state  <= sub_seq2 ;
                           cpt_seq    <= cpt_seq + 1;
           when sub_seq2 => dac_state  <= sub_seq3 ;
                           DAC_SCLK   <= '0';
           when sub_seq3 => if cpt_seq = "10000" then
                             dac_state  <= •dle ;
                           else
                             dac_state <= sub_seq0 ;
                           end if;
         end case ;
       end if;
     end process ;

   DAC_DIN <= data_decal(16);

 end behave;
```

Annex 16 : Documentation update

Version 1 : initial version

Version 2 : internal version

Version 3 : internal version

Version 4 : internal version

Version 5 (May 2006) :

Add Documentation Update annex	Add
Add board scheme	Add
Add board implantation component (top and bottom side)	Add
Modification of the description of the Command Without argument (chap. 3.2)	modify
Remove all explains about SERDES 28 bit	Delete
Modify the signal DAT(20) on the figure Output signal waveform with a 21 bits Channel link (new timing)	Modify
Add read in data format (DAQ_FEE to MERGER)	Add
Modify Annex 4 : “MERGER” FPGA synoptic with 21 bits SERDES Annex 5 : MERGER_CORE sub-module architecture Figure 8 : MERGER_DOUT_IO Module architecture Figure 10 : Merger_Shift_IO FSM module architecture	Modify
Add Annex 7 : SPARSIFICATION up LINK connector pin out Annex 8 : SPARSIFICATION down LINK connector pin out	Add
Add FPGA VHDL file description	Add
Update the DAQ matrix implementation figure	Update
Update the Group words definition figure	Update
Modify the description of the Enable to Transmit command Add a diagram	Modify
Update the Annex 1 : Logical parts block diagram with 21 bits SERDES	Update
Update FPGA pin description	Update
Add explains for Data Acquisition Transferring (chapter 5 page 8)	Add
Add Annex 2 : Power parts block diagram	Add
Update Annex 6 : Row Data Acquisition Sequence page 26	Update
Add command in chapter 3.1 Command with argument (MERGER internal register)	Add
Add command in chapter 3.2 Command without argument (LED_ON, LED_OFF, NOP)	Add
Add Annex 3 : LED interface diagram	Add
Add Annex 4 : LED command sequence	Add

Version 6 (July 2006) :

Modify the Initialization of the MERGER internal register in the chapter 3.1	Update
Modify the command without argument chapter 3.2	Update
Update Annex 3 : LED interface diagram	Update
Update Annex 4 : LED command sequence	Update

Version 7 (August 2006) :

Modify Annex 8 : MAIN and SLAVE MERGER_CORE sub-module architecture	Uodate
In Annex 8 add MAIN MERGER_CORE sub module diagram	Add
Update 8 bit and 16 bit command diagram	Update
Update FPGA pin out	Update
Update board scheme	Update
Update chapter 4.2 MERGER board acquisition time	Update
Update chapter 7.1 Internal architecture	Update
Update Annex 7 : “MERGER” FPGA synoptic with 21 bits SERDES	Update
Update VHDL file and add MERGER_DAC.vhd	Update
Update Figure 1 : command with argument (the number of clock after EN_N_ROW_BRD go high is 26)	Update
Update Figure 2 : command without argument (the number of clock after EN_N_ROW_BRD go high is 26)	Update
Add parameter “ CK_ATCMD” clock number after TRIGGER command in Figure 7 : SLAVE MERGER_CORE sub module (2/3) (equal to 2 by default for 9 clocks)	Add
Add parameter “ CK_ANCMD” clock number after normal command in Figure 8 : SLAVE MERGER_CORE sub module (3/3) (equal to 6 by default for 26 clocks)	Add

Version 8 (May 2007) :

Update Annex 13 : Board implantation components (top side)	Uodate
Update Annex 14 : Board implantation components (bottom side)	Uodate
Update Scheme page 3	Update

Table of contents